From semantic-aware to semantic-enhanced knowledge graph embedding models for link prediction

Nicolas Hubert

UNIVERSITÉ DE LORRAINE

École doctorale SIMPPE

Mesure et enrichissement sémantiques des modèles à base d'embeddings pour la prédiction de liens dans les graphes de connaissances

THÈSE

présentée et soutenue publiquement le 17 juin 2024

pour l'obtention du

Doctorat de l'Université de Lorraine

(mention Génie des systèmes industriels)

par

Nicolas Hubert

Composition du jury

Président :	Fabian Suchanek, Professor, Télécom Paris, Institut Polytechnique de Paris
Rapporteurs :	Fatiha Saïs, Professeure, Université Paris Saclay Paul Groth, Professor, University of Amsterdam
Examinateur :	Fabian Suchanek, Professor, Télécom Paris, Institut Polytechnique de Paris
Invité :	Heiko Paulheim, Professor, University of Mannheim
Directeurs :	Davy Monticolo, Professor, Université de Lorraine Armelle Brun, Professeure, Université de Lorraine

ERPI – Equipe de Recherche sur les Processus Innovatifs

Mis en page avec la classe thesul.

Acknowledgments

First and foremost, I would like to express my gratitude to all those who contributed daily to the success of this PhD. I deeply thank my supervisor Armelle for the unwavering support she provided, her numerous smiles, and good humor. Over these three years, working with Armelle has been both a real pleasure and an invaluable privilege.

I would also like to express my gratitude to my colleagues at ERPI and LORIA. In particular, I want to thank my long-time companions – Halima, Maksym, Marie, and Neal – for the moments of relaxation and camaraderie we shared. It is often forgotten that a PhD is, above all, a human adventure. Although I was physically less present at LORIA, I would like to thank Célina for our numerous discussions and mutual support. Having someone with whom to share my PhD struggles was immensely beneficial.

I would also like to thank Heiko for welcoming me into the Data and Web Science (DWS) group. The four months I spent in Mannheim were an extremely enriching and joyful experience for me, both professionally and personally. Heiko is an exceptional person. Professionally, the scope of his ideas and the depth of his thinking never cease to impress me. I am also in awe of his constant availability. I would be happy to continue collaborating with Heiko in the future. I also want to thank everyone I interacted with at the DWS group in Mannheim. Special mention goes to Adrian, Andreea, Antonis, Katharina, Martin, and Sven for lending me their university restaurant cards. Without this precious card, my attempts to discover German cuisine would have left a bitter taste. I also thank Alexander, Daniel, Rita, and Nico for their kindness towards me.

A thousand thanks also to Pierre Monnin, who played a crucial role in the success of this thesis. When I was approaching the first six months of my PhD and considering quitting to raise sheep in New Zealand, meeting Pierre transformed my research and set it on a virtuous path. Pierre has been a mentor to me, and I will be eternally grateful. I am also proud to count him as a friend and hope he continues to appreciate my quirky humor.

I also thank all those who indirectly contributed to the success of this PhD by creating an environment conducive to my well-being. Alexandre, Aurélia, Christine, Joffrey, Thomas, thank you all.

Finally, I wish to conclude these acknowledgments by thanking my family. I thank my sister – Sarah – and my parents for always being there for me. I am particularly grateful to my mother, who worked tirelessly to provide me with access to higher education and always ensured a supportive environment for my personal and academic growth. I thank her for the education and values instilled in me. I owe her a great deal of what I achieved so far.

To coffee, for fueling my working sessions and preventing naps. Less is more now, but no promises.
To Netflix, for being my loyal companion and providing hours of distraction, and yet here we are.
To sleep, which I neglected during this PhD journey. I am sorry and I will make it up to you.

Contents

List of Algorithms

List of Figures

List of Tables

Publications

This doctoral thesis is based on the following publications:

International journal articles

- Nicolas Hubert, Pierre Monnin, and Heiko Paulheim. "Beyond Transduction: A Survey on Inductive, Few Shot, and Zero Shot Link Prediction in Knowledge Graphs". *Preprint (under review).* URL: https://arxiv.org/abs/2312.04997.

- Nicolas Hubert, Pierre Monnin, Armelle Brun, and Davy Monticolo. "Sem@K : Is my knowledge graph embedding model semantic-aware?". In *Semantic Web, vol. 14, no. 6, pp. 1273-1309, 2023.* URL: https://content.iospress.com/articles/semantic-web/sw233508.

International conference articles

- Nicolas Hubert, Heiko Paulheim, Armelle Brun, and Davy Monticolo. "Do Similar Entities have Similar Embeddings?". In *The Semantic Web - 21st International Conference, ESWC 2024, Hersonissos, Crete, Greece, May 26 - May 30, 2024, Proceedings.* URL: https://arxiv.org/abs/2312.10370.

- Nicolas Hubert, Pierre Monnin, Mathieu d'Aquin, Davy Monticolo, and Armelle Brun. "PyGraft: Configurable Generation of Synthetic Schemas and Knowledge Graphs at Your Fingertips". In *The Semantic Web - 21st International Conference, ESWC 2024, Hersonissos, Crete, Greece, May 26 - May 30, 2024, Proceedings.* URL: https://arxiv.org/abs/2309.03685.

- Nicolas Hubert, Pierre Monnin, Armelle Brun, and Davy Monticolo. "Treat Different Negatives Differently: Enriching Loss Functions with Domain and Range Constraints for Link Prediction". In *The Semantic Web - 21st International Conference, ESWC 2024, Hersonissos, Crete, Greece, May 26 - May 30, 2024, Proceedings.* URL: https://arxiv.org/abs/2303.00286.

- Nicolas Hubert, Heiko Paulheim, Pierre Monnin, Armelle Brun, and Davy Monticolo. "Schema First! Learn Versatile Knowledge Graph Embeddings by Capturing Semantics with MASCHInE". In *Proceedings of the 12th Knowledge Capture Conference (K-CAP), pp. 188–196, Pensacola, FL, USA, 2023.* URL: https://dl.acm.org/doi/10.1145/3587259.3627550.

- Nicolas Hubert, Armelle Brun, and Davy Monticolo. "New Ontology and Knowledge Graph for University Curriculum Recommendation". In *Proceedings of the 21st International Semantic Web Conference (ISWC), Virtual event, 2022*. URL: https://ceur-ws.org/Vol-3254/paper349.pdf.

- Nicolas Hubert, Pierre Monnin, Armelle Brun, and Davy Monticolo. 'New Strategies for Learning Knowledge Graph Embeddings: the Recommendation Case". In *Proceedings of the 23rd International Conference on Knowledge Engineering and Knowledge Management (EKAW), Lecture Notes in Computer Science(), vol 13514, pp. 66-80, Bozen-Bolzano, Italy, 2022*. URL: https://link.springer.com/chapter/10.1007/978-3-031-17105-5_5.

International workshop articles

- Nicolas Hubert, Pierre Monnin, Armelle Brun, and Davy Monticolo. "Knowledge Graph Embeddings for Link Prediction: Beware of Semantics!". In *DL4KG 2022: Workshop on Deep Learning for Knowledge Graphs, held as part of ISWC 2022: the 21st International Semantic Web Conference, October 23 - 27, 2022*. URL: https://ceur-ws.org/Vol-3342/paper-4.pdf.

National conference articles

- Nicolas Hubert, Pierre Monnin, Mathieu d'Aquin, Davy Monticolo, and Armelle Brun. "PyGraft: un outil Python pour la génération de schémas et graphes de connaissance synthétiques". In *IC2024 : 35èmes journées francophones d'Ingénierie des Connaissances, La Rochelle, France*. URL: https://inria.hal.science/hal-04589855.

- Nicolas Hubert, Pierre Monnin, Armelle Brun, and Davy Monticolo. "Enrichissement de fonctions de perte avec contraintes de domaine et co-domaine pour la prédiction de liens dans les graphes de connaissance". In *27e Conférence Nationale en Intelligence Artificielle, La Rochelle, France*. URL: https://inria.hal.science/hal-04566996.

National workshop articles

- Nicolas Hubert, Armelle Brun, and Davy Monticolo. "Vers un système de recommandation explicable pour l'orientation scolaire". In *EGC 2022 - Extraction et Gestion des Connaissances, Workshop EXPLAIN'AI, Blois, France, 2022*. URL: https://hal.science/hal-03559471/document.

Acronyms

AI Artificial Intelligence

BK Background knowledge

BCEL Binary cross-entropy loss

BNS Bernoulli negative sampling

CNN Convolutional neural network

CWA Closed world assumption

GAN Generative adversarial network

GD Gradient descent

GNN Graph neural network

KG Knowledge graph

KGC Knowledge graph completion

KGE Knowledge graph embedding

KGEM Knowledge graph embedding model

LCWA Locally closed world assumption

LLM Large language model

LOD Linked open data

ML Machine learning

MR Mean rank

MRR Mean reciprocal rank

NLP Natural language processing

NS Negative sampling

OWA Open world assumption

OWL Web ontology language

PHL Pairwise hinge loss

PLL Pointwise logistic loss

RDF Resource description framework

RDFS Resource description framework schema

SGD Stochastic gradient descent

SCWA Stochastic closed world assumption

SPARQL SPARQL protocol and RDF query language

SW Semantic web

TCNS Type-constrained negative sampling

UNS Uniform negative sampling

URI Uniform resource identifier

URL Uniform resource locator

XML Extensible markup language

Résumé étendu

Web sémantique, ontologies et graphes de connaissances

En tant que chercheurs en informatique, nous devons manipuler un type de connaissance qui soit à la fois lisible par les machines et compréhensible par les humains. Avec le développement du *World Wide Web* (WWW, ou simplement le Web), de nombreux protocoles et normes ont été développés pour faciliter le partage de contenu sur l'Internet. Ainsi, l'information est devenue facilement accessible pour les humains. Cependant, les ordinateurs rencontrent des difficultés à traiter cette information en raison de sa grande hétérogénéité. Le *Web sémantique* (WS) vise à surmonter ce défi en fournissant des informations dans un format que les machines peuvent traiter, ainsi que les outils appropriés pour les manipuler. De plus, le WS établit des normes pour assurer l'interopérabilité et permettre le raisonnement. Dans cette thèse, nous nous intéressons particulièrement au WS, lequel enrichit les possibilités offertes par le WWW grâce aux normes établies par le World Wide Web Consortium (W3C)[1].

Le WS repose sur un ensemble structuré de technologies clés. En particulier, le Resource Description Framework (RDF)[2] constitue le format d'échange standard en offrant un cadre pour exprimer les informations et, surtout, les relations entre elles. Il étend la structure de lien du Web en utilisant des Uniform Resource Identifers (URIs) pour nommer les entités et leurs relations. Le modèle de données du RDF repose sur l'idée d'utiliser ces URIs pour former des déclarations sous la forme de *triplets*. Un triplet se présente sous la forme (`sujet`, `prédicat`, `objet`).

Un graphe RDF est défini comme un ensemble de tels triplets. Les `sujets` et `objets` sont les sommets du graphe, tandis que le `prédicat` est représenté schématiquement comme un arc dirigé reliant un `sujet` à un `objet` (voir Fig. 1).

Bien que le RDF soit puissant pour représenter les ressources, il est intrinsèquement limité car il ne peut pas exprimer des structures complexes telles que les classes, les héritages et certaines propriétés des relations. La modélisation de ces types de structures plus complexes est l'objectif des *ontologies*. En essence, une ontologie peut être définie comme une spécification explicite d'une conceptualisation. Une ontologie définit les concepts principaux et les relations d'un domaine d'intérêt, ainsi que les règles régissant leurs interactions [58]. Un exemple simplifié d'une ontologie décrivant le domaine des publications de recherche est illustré en Fig 2. La création d'ontologies ne se limite pas à l'utilisation de RDF : d'autres langages tels que le RDF Schema (RDFS) et le Web Ontology Language (OWL) apportent des fonctionnalités avancées pour exprimer des structures ontologiques plus complexes.

Passant d'une structure conceptuelle à une représentation concrète, nous évoluons des ontologies vers les graphes de connaissances. Autrement dit, la transition des ontologies aux *graphe de connaissances* marque un pas de l'abstraction théorique vers l'application pratique et l'interaction

[1] `https://www.w3.org/`
[2] `https://www.w3.org/RDF/`

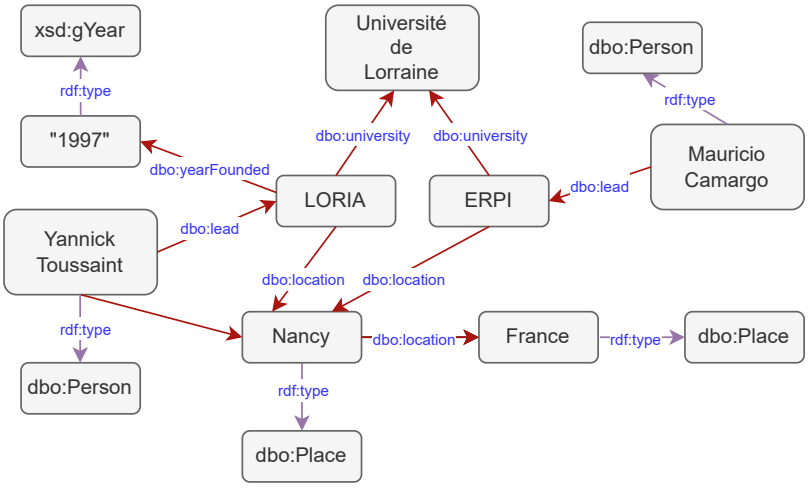


Figure 1: Exemple d'un graphe RDF

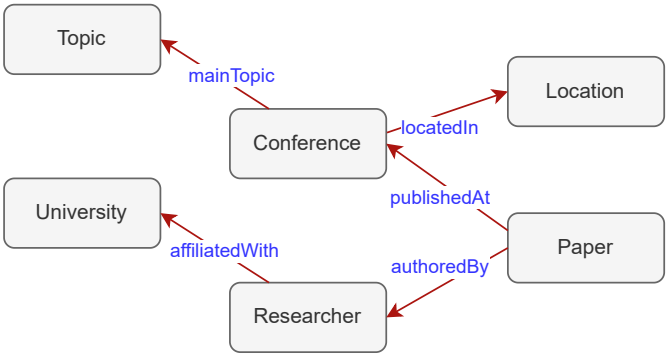


Figure 2: Exemple visuel d'une ontologie

avec des données réelles. Tandis que les ontologies fournissent un cadre pour définir et organiser les connaissances, les graphes de connaissances sont des représentations concrètes de ces connaissances.

Bien que le terme graphe de connaissances soit largement accepté, sa définition formelle fait encore l'objet de débats dans la communauté de recherche. La notion de *graphe de connaissances* englobe plusieurs interprétations; sa définition n'est pas univoque.

Dans ce manuscrit, nous utilisons la définition inclusive proposée par Hogan *et al.* [69] :

Definition 1 (Graphe de Connaissances) *Un graphe de connaissances est un graphe de données visant à accumuler et transmettre la connaissance du monde réel, dont les nœuds représentent des entités d'intérêt et dont les arêtes représentent les relations entre ces entités.*

Après avoir défini les graphes de connaissances et exploré leur rôle dans la représentation structurée de données complexes, il est essentiel de reconnaître que ces graphes ne sont pas statiques, mais évoluent et s'enrichissent continuellement. Cette dynamique soulève un défi crucial : comment pouvons-nous étendre et enrichir efficacement ces graphes avec de nouvelles connexions et relations ? C'est ici que la *prédiction de liens* entre en jeu, en tant que méthode clé pour découvrir et intégrer de nouvelles connaissances au sein de graphes existants.

Prédiction de liens dans les graphes de connaissances

La *prédiction de liens* est une composante fondamentale dans l'évolution et l'enrichissement des graphes de connaissances. Elle répond à une question centrale : étant donné l'état actuel d'un graphe, quelles nouvelles relations peuvent être inférées ou prédites pour compléter et étendre notre compréhension du domaine représenté ? Cette tâche devient encore plus pertinente dans le contexte de l'immense volume de données disponibles, souvent incomplètes ou en constante évolution.

Bien qu'une grande quantité de données relationnelles soit publiquement disponible, les graphes de connaissances restent intrinsèquement incomplets en raison de plusieurs facteurs, découlant de leur processus de construction, des hypothèses sur le monde qu'ils représentent, et de la nature de l'information qu'ils contiennent.

La prédiction de liens se concentre spécifiquement sur la prédiction de l'existence d'un lien (relation) entre deux entités. En se basant sur une entité donnée et une relation existante, l'objectif est de prédire l'entité manquante. Plus formellement, la prédiction de lien consiste à exploiter les faits existants dans un graphe de connaissances pour en inférer de nouveaux. Cela revient à prédire l'entité réelle pour $(?, r, t)$ (prédiction de la tête) et $(h, r, ?)$ (prédiction de la queue).

Plus spécifiquement, nous limitons notre discussion à la prédiction de lien dans le cadre *transductif*, c'est-à-dire que toutes les entités candidates dans les triplets à prédire sont déjà observées dans le graphe de connaissances. Cela diffère du cadre *inductif*, où de nouvelles entités ou relations peuvent émerger au moment des tests.

En fin de compte, la tâche de prédiction de lien peut être abordée avec différentes approches, par exemple des modèles basés sur la probabilité, sur les chemins, sur les règles et sur les embeddings [32].

Cette thèse s'intéresse à des modèles qui reposent sur des techniques d'apprentissage statistique. Ces modèles génèrent des représentations vectorielles des entités et des relations, connues sous le nom d'*embeddings*. Ces embeddings peuvent alors êtres utilisés à différentes fins, notamment dans le cadre de l'élaboration d'un système de recommandation de parcours scolaires

(section suivante).

Recommandation de parcours scolaires

Cette thèse s'inscrit dans le cadre du projet AILES[3] (Accompagnement à l'intégration des Lycéens dans l'Enseignement Supérieur). Le projet AILES vise les lycéens, les néo-étudiants, leurs familles, mais aussi l'ensemble des acteurs de l'éducation et de l'orientation. Il associe trois Universités (Université de Reims Champagne-Ardenne, Université de Lorraine, Université de technologie de Troyes) et deux Rectorats (Nancy-Metz et Reims). Il s'agit d'une opération soutenue par l'Etat dans le cadre de l'action « Territoires d'innovation pédagogique » du programme d'investissement d'avenir.

Dans ce contexte, un des piliers de cette thèse est d'effectuer de la recommandation de parcours scolaires aux lycéens et aux étudiants jusqu'à la licence.

Les systèmes de recommandation dans le contexte des parcours scolaires utilisent des données sur les intérêts, les performances académiques et les objectifs professionnels des étudiants pour suggérer des cours, des spécialisations ou des programmes d'études [180, 127, 125]. Ils s'appuient principalement sur deux techniques :

- Filtrage collaboratif : Cette méthode fait des recommandations basées sur les similarités entre les utilisateurs. Si un étudiant A a suivi et apprécié un ensemble de cours similaires à un étudiant B, alors les cours appréciés par B mais pas encore suivis par A peuvent être recommandés à A.
- Filtrage basé sur le contenu : Ici, les recommandations sont faites sur la base des caractéristiques des cours ou des programmes d'études (comme le domaine, le niveau de difficulté, les compétences visées) et des intérêts ou des performances de l'étudiant.

Ces systèmes, en analysant les données disponibles, peuvent aider à personnaliser le parcours éducatif des étudiants, en suggérant des options qui correspondent à leurs profils et aspirations.

L'utilisation d'ontologies pour la recommandation de parcours scolaires, en parallèle ou en complément du filtrage basé sur le contenu, présente un potentiel significatif pour améliorer la personnalisation et la précision des recommandations [81, 82]. Dans un contexte de recommandation de parcours scolaires, le filtrage basé sur le contenu repose sur l'analyse des caractéristiques des cours et des programmes (comme le domaine, le niveau de difficulté, les compétences visées) ainsi que sur les intérêts et les performances des étudiants. Cela permet de recommander des options éducatives qui semblent pertinentes sur papier, mais cela peut manquer de profondeur et de contextualisation. Les ontologies, en revanche, offrent une structure de données riche qui permet de modéliser de manière détaillée les connaissances et compétences associées à chaque cours ou programme, ainsi que leur interrelation avec les carrières, les intérêts et d'autres cours. Cela peut aider à créer des recommandations plus nuancées qui tiennent compte non seulement de la surface des intérêts de l'étudiant mais aussi de leurs objectifs à long terme et de la progression logique des compétences [81, 82].

Dans cette thèse nous cherchons à aller au-delà de ces approches en tirant pleinement parti des possibilités inhérentes à la fois aux ontologies *et* aux graphes de connaissance. Leur intégration dans les systèmes de recommandation scolaire offre une évolution significative en termes de personnalisation, de précision des recommandations, mais aussi d'explicabilité [180, 127, 72].

En l'occurrence, les ontologies peuvent permettre de modéliser les connaissances et compétences associées à chaque cours ou programme, facilitant ainsi la recommandation de parcours

[3] `https://projetailes.com/`

éducatifs qui sont non seulement adaptés aux intérêts et performances actuels de l'étudiant mais aussi alignés sur les compétences requises dans leur domaine de carrière souhaité [125, 33]. Quant aux graphes de connaissance, ils permettent de connecter diverses entités (comme les cours, les compétences, les carrières, et les intérêts) [180]. En utilisant des graphes de connaissance, le système de recommandation peut comprendre les liens complexes entre différents éléments du curriculum et comment ils s'alignent sur les objectifs de carrière ou les préférences personnelles des étudiants.

L'application de ces technologies permet de passer d'une approche basée sur les tendances générales et les préférences superficielles à une méthode enracinée dans une connaissance plus fine du contexte scolaire.

1 Motivations et questions de recherche

L'étude des modèles à base d'embeddings pour la prédiction de liens dans les graphes de connaissance est un champ actif dans la recherche contemporaine en apprentissage automatique. Ces modèles sont connus pour leur compétence à capturer les complexités et nuances des entités et de leurs interrelations. De plus, ils reposent sur l'idée principale que les représentations vectorielles apprises encodent la sémantique latente d'un graphe de connaissances. Par conséquent, les embeddings appris par ces modèles sont exploités dans plusieurs applications réelles [89].

Malgré ces avancées, les modèles à base d'embeddings rencontrent souvent des contraintes significatives découlant de leur nature purement basée sur l'apprentissage automatique. En effet, l'hypothèse susmentionnée selon laquelle les embeddings sont capables d'encoder correctement la sémantique des KG a été récemment remise en question [86]. Cela suggère que les modèles relationnels appris à l'aide de méthodes purement statistiques ne suffisent pas à encoder les entités et relations de manière à ce que leurs représentations puissent être utilisées dans des tâches où la sémantique est importante. En conséquence, des travaux récents ont étudié l'utilisation d'approches neuro-symboliques pour apprendre des embeddings [111, 35, 87]. Ces approches vont au-delà de la simple utilisation de triplets relationnels décrivant les relations entre entités contenus dans le graphe. Au lieu de cela, elles exploitent également la richesse des connaissances structurées contenues dans les ontologies et injectent ces connaissances lors de l'entraînement, afin que les embeddings s'alignent mieux avec la signification et le rôle incarnés par les entités et relations.

Cependant, pour développer de telles approches neuro-symboliques, il est nécessaire de disposer de ressources sémantiquement riches, en particulier des jeux de données sous-tendus par un schéma ou une ontologie. Malheureusement, dans certaines situations, de telles ressources ne sont pas publiquement accessibles. Considérant la tâche fondamentale de prédiction de lien, nous observons un manque considérable de jeux de données basés sur des schémas, notamment en ce qui concerne les *benchmarks* les plus populaires. De plus, les ressources sémantiquement riches ne sont pas facilement accessibles pour des applications dans certains domaines particuliers. Etant donné que cette thèse se concentre sur les applications éducatives, nous déplorons une pénurie d'ontologies et de graphes de connaissances modélisant le domaine éducatif, et plus particulièrement la période de transition du lycée à l'université. De manière générale, le manque de ressources sémantiques dans des contextes spécifiques entrave le développement d'approches neuro-symboliques, ce qui explique pourquoi la plupart des travaux récents sont dans la grande majorité représentés par des approches uniquement basées sur de l'apprentissage statistique [170].

Les modèles à base d'embeddings – qu'ils soient entraînés uniquement avec des techniques d'apprentissage automatique ou avec une approche neuro-symbolique – sont principalement mo-

tivés par un seul objectif : améliorer la performance en matière de prédiction de liens au regard de métriques basées sur le rang [144]. Bien que ces métriques fournissent une mesure quantifiable de la performance, elles offrent une perspective plutôt unidimensionnelle de l'efficacité des modèles à base d'embeddings. Cette approche se concentre principalement sur la capacité d'un modèle à assigner de forts scores de plausibilité au triplet *ground-truth*, ce qui, bien qu'important, n'est qu'un aspect de leur fonctionnalité dans des applications réelles. Par exemple, dans le contexte des systèmes de recommandation, et plus particulièrement dans l'orientation scolaire, il n'est pas seulement essentiel que la formation la plus pertinente au regard du profil-élève soit recommandée en haut de la liste de recommandations : il est également crucial que la l'entièreté de la liste de cursus scolaires proposés soit alignée avec le profil, les compétences et les aspirations de l'élève. En élargissant et diversifiant le spectre des formations pertinentes proposées, l'élève peut alors explorer par lui-même l'ensemble des choix qui s'offrent à lui. En résumé, l'accent mis sur l'évaluation basée sur le rang néglige l'évaluation de certaines dimensions qualitatives des modèles à base d'embeddings, dont l'importance est réelle. Par exemple, les métriques traditionnellement utilisées n'évaluent pas la capacité des modèles à générer des prédictions qui sont sémantiquement correctes en dehors du score assigné au triplet *ground-truth*. De plus, le paradigme d'évaluation contemporain échoue souvent à considérer les aspects qualitatifs des embeddings générés, tels que leur interprétabilité, la richesse de leur représentation sémantique et leur alignement avec la cognition et la compréhension humaines.

Cette thèse s'appuie sur l'état actuel de la recherche sur les modèles à base d'embeddings et identifie trois limitations principales :

- **Manque de ressources sémantiquement riches**. L'une des principales limitations est la rareté des jeux de données enrichis de contenu sémantique. De nombreux jeux de données existants manquent de profondeur et de variété en termes d'informations disponibles pour entrainer des modèles neuro-symboliques. Cet écart entrave la capacité des modèles à base d'embeddings à tirer pleinement parti du pouvoir expressif des graphes de connaissance, limitant ainsi leur potentiel dans des applications nécessitant une compréhension nuancée et une interprétation des données.

- **Cadre d'évaluation unidimensionnel**. Les méthodes d'évaluation existantes, qui se concentrent largement sur les métriques basées sur le rang [144], offrent une vue limitée de la performance d'un modèle. Bien que ces métriques mesurent efficacement la précision de la prédiction de liens, elles ne tiennent pas compte d'autres aspects cruciaux tels que la richesse sémantique et la pertinence contextuelle des liens prédits. Ce cadre d'évaluation unidimensionnel conduit à une compréhension incomplète des capacités des modèles et néglige leur nature multivariée.

- **Manque de considérations sémantiques dans les approches basées sur l'apprentissage automatique**. Les modèles actuels, étant basés sur les principes de l'apprentissage automatique, fonctionnent principalement sur l'optimisation numérique et la reconnaissance de motifs. Cette approche, bien qu'efficace dans le traitement de données à grande échelle et l'identification de motifs, ne capture pas intrinsèquement les nuances sémantiques et les subtilités contextuelles présentes dans les graphes de connaissance. L'absence d'une approche neuro-symbolique, qui combine l'intelligence artificielle symbolique avec des réseaux neuronaux, signifie que les modèles sont limités à ce qui peut être évalué quantitativement, négligeant souvent les couches sémantiques qualitatives des données.

Compte tenu des limitations discutées ci-dessus, nous considérons donc les questions de recherche suivantes :

QR1. Comment proposer des ressources publiquement accessibles et sémantiquement riches pour favoriser le développement d'approches neuro-symboliques pour des applications basées sur des graphes de connaissance, notamment pour la prédiction de liens ?

QR2. Étant donné la dépendance exclusive aux métriques basées sur le rang pour évaluer les modèles à base d'embeddings, pouvons-nous proposer de nouvelles métriques qui évalueraient qualitativement différents aspects de ces modèles, en particulier leur capacité à prédire des liens sémantiquement plausibles ?

QR3. Sur la base de telles métriques et en utilisant des jeux de données sémantiquement riches comme précédemment proposé, pouvons-nous concevoir des approches d'entraînement neuro-symboliques pour exploiter efficacement la richesse des connaissances disponibles ?

2 Contributions

L'objectif de cette thèse doctorale est de répondre à ces limitations en contribuant à la création de ressources sémantiquement riches (notamment des graphes de connaissances) et en explorant des approches pour améliorer la compréhension et l'évaluation sémantique des modèles à base d'embeddings. Plus spécifiquement, les contributions de cette thèse sont les suivantes :

1. L'enrichissement et la publication de versions sémantiquement riches de jeux de données courants pour la prédiction de liens [79, 80] (**QR1**).
2. La création, le développement et la publication d'une ontologie (EducOnto) et d'un graphe de connaissances (EduKG) dans le domaine de l'orientation scolaire [72, 71] (**QR1**).
3. Le développement, la maintenance et la publication de PyGraft : un outil Python *open-source* pour générer des ontologies et des graphes de connaissances synthétiques sur la base d'un large ensemble de paramètres définis par l'utilisateur [77] (**QR1**).
4. Une nouvelle métrique sémantique pour évaluer la capacité sémantique des modèles à base d'embeddings : Sem@K [76, 75, 79] (**QR2**).
5. Une réévaluation complète des modèles à base d'embeddings, à l'aune de leurs capacités sémantiques [79] (**QR2**).
6. Le développement subséquent d'approches neuro-symboliques qui sont non seulement évaluées sur la base de leur capacité sémantique mais qui intègrent également des informations sémantiques pendant l'entraînement pour améliorer leurs capacités prédictives (métriques basées sur le rang) [78, 80] (**QR3**).
7. Une étude poussée des approches neuro-symboliques et de leur usage pour un certain nombre d'applications, notamment dans le cadre des systèmes de recommandation [76] (**QR3**).

Résumé des chapitres et des contributions

Dans ce manuscrit, le Chapitre 2 offre une connaissance approfondie des modèles à base d'embeddings pour les graphes de connaissance. Ce chapitre constitue un socle fondamental de concepts clés et de terminologies, établissant ainsi la base pour une appréciation plus profonde des contributions détaillées à partir du Chapitre 3. Chaque chapitre est ensuite dédié à la présentation

des contributions de cette thèse et suit une approche structurée : le contexte de recherche et les motivations sont premièrement établies. Les limites de l'état de l'art actuel sont identifiées, menant à la formulation de questions de recherche spécifiques. Ces questions de recherche dites secondaires nous permettent de répondre aux questions de recherche plus larges (RQ1, RQ2 et RQ3) présentées dans la section précédente. Les contributions de chaque chapitre sont ensuite décrites, toujours au regard de la thématique générale de cette thèse.

Dans l'ensemble, la structure et le contenu de ce manuscrit sont comme suit :

Le Chapitre 2 se concentre sur la tâche de prédiction de liens dans les graphs de connaissance à l'aide de modèles à base d'embeddings. Cela implique de définir en premier lieu la notion de graphe de connaissances. Différentes conceptions existent, et nous y expliquons clairement les raisons de notre choix. Ce chapitre commence par établir le cadre général de la prédiction de liens puis aborde des points plus pointus et techniques autour de la définition, de l'entraînement et de l'évaluation des modèles à base d'embeddings. Enfin, nous concluons sur un panorama des modèles à base d'embeddings utilisés au cours de cette thèse.

Les premières contributions de cette thèse sont présentées dans le Chapitre 3. En particulier, il est fait mention du manque de ressources sémantiquement riches pour un grand nombre d'applications utilisant des graphes de connaissances. La première contribution cible la tâche de prédiction de lien et consiste en l'enrichissement sémantique et la mise à disposition de versions augmentées des jeux de données couramment utilisés. Ensuite, le manque de ressources publiquement disponibles dans le domaine de l'orientation scolaire pour effectuer des recommandations de cursus est abordé. Plus spécifiquement, EducOnto et EduKG sont présentés en détail comme des ressources adéquates pour mener des recherches à l'intersection des systèmes de recommandation et de l'intelligence artificielle explicable en éducation. Enfin, le champ d'application du Chapitre 3 est volontairement élargi. Nous mettons d'abord en lumière le besoin d'une collection plus large et plus diversifiée de jeux de données *benchmarks* pour des applications basées sur des graphes de connaissances, et soulignons la rareté relative des jeux de données disposant d'un schéma pour de telles applications. Ces problématiques nous ont menés à rendre PyGraft publiquement accessible. PyGraft est un outil *open-source* en Python pour générer des ontologies et des graphes de connaissances synthétiques. L'outil est doté d'un ensemble étendu de paramètres que l'utilisateur peut contrôler.

Fournir un outil pour générer des jeux de données sous-tendus par un schéma a notamment pour but de faciliter le développement d'approches neuro-symboliques. La richesse de connaissances sémantiques supplémentaires contenues dans de tels schémas peut ainsi être exploitée pour améliorer la précision des modèles à base d'embeddings pour la tâche de prédiction de liens. Cependant, dans le Chapitre 4, nous soulevons un certain nombre de préoccupations essentielles quant à l'évaluation de tels modèles dans la littérature. En particulier, alors même que certains travaux utilisent des schémas comme source d'informations supplémentaires, les modèles à base d'embeddings sont uniquement évalués par rapport à des métriques basées sur le rang. En se basant sur les limites existantes de telles métriques, nous fournissons également un argumentaire autour d'une procédure d'évaluation plus holistique. Nous contribuons à cet effort en proposant Sem@K, une nouvelle métrique sémantique qui se concentre sur la conformité des prédictions avec le domaine et le co-domaine des relations d'un graphe de connaissances. Sem@K existe sous différentes formes en fonction de la nature du jeu de données. Nous réévaluons ensuite les modèles à base d'embeddings populaires selon leur capacité sémantique à l'aune de Sem@K. Cette étude complète fournit un regard neuf sur ces modèles, avec certaines tendances communes entre modèles d'une même famille. En particulier, une analyse approfondie montre qu'un modèle peut être performant au regard des métriques basées sur le rang, mais procurer des performances médiocres en termes de Sem@K, et *vice-versa*. Ce résultat doit nous conduire à adopter une

vigiliance accrue lors de l'utilisation de ces modèles dans des applications réelles. Nous fournissons un exemple concret de l'utilité de Sem@K dans le contexte spécifique des systèmes de recommandation.

Le Chapitre 5 va un pas au-delà de la simple évaluation des capacités sémantiques des modèles à base d'embeddings. Maintenant que nous disposons d'une métrique pour mesurer cette dimension, nous étudions comment nous pouvons améliorer la capacité sémantique d'un modèle dans le contexte de la prédiction de liens. Cela nous amène à concevoir des modèles sémantiquement améliorés qui, contrairement aux modèles agnostiques, exploitent des informations basées sur le schéma dans leur procédure d'entraînement. Plus spécifiquement, nous proposons une manière d'intégrer la connaissance du domaine et co-domaine des relations dans les fonctions de perte populaires pour la prédiction de liens. L'approche que nous proposons améliore considérablement la capacité sémantique des modèles étudiés pour la prédiction de liens, comme en témoigne l'augmentation notable des valeurs de Sem@K. En outre, dans la plupart des cas, nous observons également une amélioration des performances par rapport aux métriques basées sur le rang. Cette contribution souligne l'importance des approches neuro-symboliques. En nous appuyant sur ces résultats prometteurs, nous adoptons ensuite un point de vue différent : au lieu d'enrichir les fonctions de perte, nous optons pour l'exploitation des connaissances sur les classes et les héritages de sous-classes pour éviter l'initialisation aléatoire des embeddings et, à la place, les initialiser intelligemment selon le schéma. Bien que cette approche soit simple, elle augmente significativement les capacités sémantiques des modèles étudiés. Notamment, les embeddings résultants sont vraisemblablement mieux en phase avec la sémantique globale des graphes, comme en témoigne leur polyvalence leur potentielle réutilisation dans d'autres tâches telles que la classification de nœuds et le regroupement d'entités. Ce nouveau résultat – en plus de souligner les avantages des approches neuro-symboliques – ouvre de nombreuses directions de recherche potentielles autour de la notion même d'embeddings.

Enfin, le Chapitre 6 résume les contributions principales de cette thèse et ouvre des perspectives de recherche futures.

Chapter 1

Introduction

Contents

The notion of *embeddings* is a cornerstone of machine learning models, as they underpin a multitude of applications across the AI spectrum. Embedding-based models translate high-dimensional data, like text or images, into a lower-dimensional space. For instance, in the context of language, embeddings transform words, sentences, or even entire documents into vectors of real numbers [109]. This transformation is not merely a compression of data, but a sophisticated representation that captures the semantic and syntactic nuances of the language. The essence of these representations is to encapsulate the inherent meaning and semantics in a format that is amenable to computational processes.

One specific application this thesis is interested about is in the realm of knowledge graph embedding models (KGEMs). These models extend the idea of embeddings to the structured data representation paradigm of knowledge graphs (KGs). A knowledge graph is a graph-based network that represents entities and their interrelations in the form of triples (h, r, t), where h and t denote the head and tail entities of the triple, whereas r denotes the semantic property holding between these entities. KGs can be seen as a modern data representation paradigm and respond to the increasing need for storing, linking and making sense of data. Importantly, KGs are often underpinned by an ontology, which can be defined as a set of concepts and categories in a given domain, along with properties and constraints that clearly define how these concepts are classified into categories, and how they are supposed to interact between themselves.

By applying the concept of embeddings to KGs, these models learn to represent the nodes (entities) and edges (relationships) of a graph as dense vectors. This transformation is crucial as it enables the encoding of relational and semantic information inherent in the graph into a format that can be effectively utilized in machine learning (ML) tasks.

However semantically rich and useful these KGs are in knowledge-intensive applications, it is essential to recognize that KGs are inherently incomplete [69]: no single KG can encompass the entirety of human knowledge or represent the evolving nature of information fully.

This brings us to the heart of our journey: link prediction (LP) in KGs. This fundamental and crucial task aims at inferring missing connections within KGs. In essence, link prediction is concerned with completing the knowledge contained in KGs by predicting links between entities. By developing models that can identify and suggest such links, the intent is to complete the

vast tapestry of knowledge captured in these graphs. While several approaches exist to perform LP [48, 18, 147], this thesis focuses more specifically on the use of the aforementioned KGEMs, which tend to be the most effective and commonly used approach in LP [144].

The significance of embedding the constituents of a KG lies not only in the possibility to infer new links, but also in the enhanced ability to perform a variety of downstream applications. For instance, embedded representations of KGs can dramatically improve the performance of recommendation systems (RSs) [59] and facilitate more effective information retrieval (IR) [53]. In essence, embeddings open a plethora of possibilities for harnessing the wealth of information contained within KGs.

In this thesis, more particularly, we are interested in providing curricula recommendations to students using such KGEMs. Therefore, we intend to instantiate the LP task with a specific objective: instead of inferring any kind of links, we define a *target* link which, in our case, materializes the recommendation to study in a given educational institution.

To help develop this recommender system, this thesis work is supported by the project AILES (Support for the integration of high school students into higher education). The AILES project targets high school students, newly enrolled students, their families, as well as all education and guidance stakeholders. It involves three French universities (University of Reims Champagne-Ardenne, University of Lorraine, University of Technology of Troyes) and two educational authorities (Nancy-Metz and Reims).

In this context, one of the main pillars of this project is to provide guidance on academic pathways to high school students and students up to the bachelor's degree level.

Recommendation systems in the context of academic pathways utilize data on students' interests, academic performance, and career goals to suggest courses, specializations, or study programs [180, 127, 125]. They primarily rely on two techniques:

- Collaborative filtering: This method provides recommendations based on similarities between users. If student A has taken and enjoyed a set of courses similar to those of student B, then the courses appreciated by B but not yet taken by A can be recommended to A.

- Content-based filtering: Here, recommendations are made based on the characteristics of courses or study programs (such as domain, difficulty level, targeted skills) and the student's interests or performance. By analyzing available data, these systems can help personalize students' educational journeys by suggesting options that align with their profiles and aspirations.

The use of ontologies for academic pathway recommendation, alongside or complementing content-based filtering, holds significant potential to enhance the personalization and accuracy of recommendations [81, 82]. In the context of academic pathway recommendation, content-based filtering relies on analyzing the features of courses and programs (such as domain, difficulty level, targeted skills), as well as students' interests and performance. This allows for recommending educational options that may seem relevant on paper but may lack depth and contextualization. On the other hand, ontologies offer a rich data structure that allows for detailed modeling of the knowledge and skills associated with each course or program, as well as their interrelation with careers, interests, and other courses. This can help create more nuanced recommendations that take into account not only the surface-level interests of the student but also their long-term goals and the logical progression of skills [81, 82].

In this thesis, we aim to go beyond these approaches by fully leveraging the inherent possibilities of both ontologies *and* knowledge graphs. Their integration into academic recommendation systems offers a significant advancement in terms of personalization, recommendation accuracy,

and explainability [180, 127, 72].

Specifically, ontologies can model the knowledge and skills associated with each course or program, thereby facilitating the recommendation of educational pathways that are not only tailored to the student's current interests and performance but also aligned with the skills required in their desired career field [125, 33]. As for knowledge graphs, they enable the connection of various entities (such as courses, skills, careers, and interests) [180]. By utilizing knowledge graphs, the recommendation system can understand the complex links between different curriculum elements and how they align with career goals or students' personal preferences.

The application of these technologies allows for a shift from an approach based on general trends and superficial preferences to a method rooted in a finer understanding of the academic context.

1.1 Motivations and research questions

The realm of KGEMs for LP is an active field in current ML research. These models, known for their proficiency in capturing the complexities and nuances of entities and their interrelationships, stand at the forefront of efforts to integrate structured knowledge into a format that is computationally manageable. In addition, these models rely on the main idea that the learned vector representations encode the overall semantics of the KG. Consequently, embeddings learned by KGEMs are leveraged in several real-world applications [89].

Despite these advancements, KGEMs often encounter significant constraints stemming from their purely machine learning-based nature. In fact, the aforementioned assumption that embeddings are able to properly encode KG semantics has been challenged recently [86]. This suggests that the relational patterns learned using purely statistical methods are not sufficient to encode entities and relations in such a way that their representations can be utilized in tasks where semantics matters [74]. Consequently, recent works have investigated the use of neuro-symbolic approaches to learn higher-qualities embeddings [111, 35, 87]. These approaches go beyond the mere use of relational triples describing the relations between individuals contained in the KG. Instead, they also leverage the wealth of structured knowledge encountered in ontologies and inject this background knowledge (BK) during training, so that embeddings can align better with the profound meaning and role embodied by entities and relations.

However, to develop neuro-symbolic approaches (*i.e.*, approaches combining neural architectures with symbolic knowledge), there is a need for semantically rich resources, especially datasets that are underpinned by a proper and readily available schema or ontology. From a high-level viewpoint, schemas and ontologies can be seen as blueprints specifying how general concepts are expected to interact together, and under which constraints. Unfortunately, there is a severe lack of such resources in some situations. Considering the fundamental task of LP, we observe a paucity of schema-based datasets, as evidenced by the fact that the most popular benchmarks do not originally come with schema-based information. In addition, semantically rich resources are not easily accessible for particular domain applications. As this thesis is concerned with the specific application of curricula recommendations to high schoolers and students, we face a paucity of ontologies and KGs modelling the choices of academic curricula. Arguably, the general lack of semantic resources in specific contexts hinders the development of neuro-symbolic approaches, which explains why most recent works, *e.g.* new KGEM proposals, are still overly represented by pure ML-based approaches [170].

KGEMs – whether trained with only ML or with a neuro-symbolic approach – are still predominantly motivated by one single objective: improving KGEM performance w.r.t. rank-

based metrics such as Hits@K, Mean Rank (MR), and Mean Reciprocal Rank (MRR) [144]. While these metrics provide a quantifiable measure of performance, they tend to offer a rather unidimensional perspective on the effectiveness of KGEMs. This approach primarily focuses on how well the model predicts or ranks relationships within the KG, which, although important, is just one aspect of their functionality and usefulness in real-world applications. For instance, in the context of e-commerce RSs, it is not only essential that the best item is recommended at the top of the ranked list of recommendations: it is also crucial that the whole list of recommended items is aligned with the purchasing intent of the user.

Additionally, in this thesis we claim that this evaluation paradigm often fails to consider the qualitative aspects of the generated embeddings, such as their interpretability, the richness of the semantic representation, and their alignment with human cognition and understanding.

To summarize, this thesis builds on the presented current state of research of KGEMs and identifies three primary limitations:

- **Lack of semantically rich resources**. One of the primary limitations is the scarcity of resources that are rich in semantic content tailored for the effective application of KGEMs. Many existing datasets lack the depth and variety of semantic information necessary to train models to understand and replicate complex real-world scenarios. This gap hinders the ability of KGEMs to fully leverage the power of KGs, thus limiting their potential in applications that require nuanced understanding and interpretation of data.

- **Unidimensional evaluation framework**. The existing evaluation methods, which focus largely on rank-based metrics, offer a limited view of a model's performance. While these metrics efficiently measure the accuracy of link prediction, they fail to account for other vital aspects such as the semantic richness and contextual relevance of the predicted links. This unidimensional evaluation framework leads to an incomplete understanding of the models' capabilities and overlooks the multifaceted nature of KGs.

- **Lack of semantic considerations in machine learning-based approaches**. Current models, being grounded in machine learning principles, primarily operate on numerical optimization and pattern recognition. This approach, while effective in handling large-scale data and identifying patterns, does not inherently capture the semantic nuances and contextual intricacies present in KGs. The absence of a neuro-symbolic approach means that the models are limited to what can be quantitatively assessed, often neglecting the qualitative, semantic layers of the data.

Given the limitations discussed above, we therefore consider the following research questions:

RQ1. How can we propose publicly available and semantically rich resources to foster the development of neuro-symbolic approaches for KG-based applications, *e.g.* link prediction?

RQ2. Given the sole reliance on rank-based metrics for evaluating KGEMs, can we propose new metrics that would qualitatively assess different aspects of these models, especially their semantic awareness?

RQ3. On the basis of such aforementioned metrics and using semantically rich datasets as previously proposed, can we design neuro-symbolic training approaches to effectively leverage the wealth of available symbolic knowledge?

1.2 Research contributions

The objective of this doctoral thesis is to address these limitations by contributing to the curation of semantically rich resources (*e.g.* KGs) and exploring approaches to enhance the semantic understanding and evaluation of KGEMs. This involves:

- Building and making both domain-specific and domain-agnostic KGs publicly available to foster research at the intersection of machine learning and symbolism.
- Developing evaluation metrics and methodologies that go beyond rank-based assessments, aiming to capture the semantic and contextual accuracy of KGEMs.
- Investigating the integration of neuro-symbolic approaches to enrich the models' understanding of semantic nuances in KGs.
- Evaluating the impact of these enhancements on the versatility and adaptability of the models, with the goal of producing representations that are not task-specific but broadly applicable.

More specifically, the contributions of this work are the following (where we specify in parenthesis which research question the contribution refers to):

1. The enrichment and release of semantically rich versions of mainstream datasets for link prediction [79, 80] (**RQ1**).
2. The curation, development, and release of an ontology (EducOnto) and KG (EduKG) in the educational domain [71] (**RQ1**).
3. The development, maintenance, and release of PyGraft: an open-source Python tool for generating synthetic ontologies and KGs on the basis of a large set of user-defined parameters [77] (**RQ1**).
4. Sem@K, a novel semantic-oriented metric for assessing the semantic awareness of KGEMs [76, 75, 79] (**RQ2**).
5. A comprehensive reassessment of KGEMs in light of their semantic capabilities [79] (**RQ2**).
6. The subsequent development of neuro-symbolic approaches that are not only evaluated based on their semantic capability but also incorporate semantic information during training to enhance their predictive abilities [78, 80] (**RQ3**).
7. An investigation into how such neuro-symbolic approaches can be leveraged for downstream tasks. Our focus is on recommender systems and how these neuro-symbolic training approaches can enhance the quality of recommended items [76] (**RQ3**).

1.3 Thesis outline

In this manuscript, Chapter 2 provide extensive background knowledge about KGs and KGEMs. This foundational groundwork is crucial for ensuring that key concepts and terminology are well-understood, setting the stage for a deeper appreciation of the contributions detailed from Chapter 3 onwards. Each chapter dedicated to presenting the contributions of this thesis follows a structured approach: it begins by setting the context and outlining the motivations. This

is followed by an identification of the limitations in the current state-of-the-art, leading to the formulation of specific sub research questions. These sub research questions are instrumental in addressing the broader research questions RQ1, RQ2, and RQ3, as delineated in Section 1.1. The contributions of each chapter are then outlined, contributing uniquely to the overarching themes of the thesis. Overall, the structure and content of this manuscript can be summarized as follows:

Chapter 2 is concerned with KGEMs and the link prediction task, which involves defining KGs in the first place. After presenting the competing definitions of KGs and explaining the rationale of our choice, this chapter sets up the framework for link prediction and subsequently dives into the intricacies of defining, training, and evaluating KGEMs. Finally, a review of the models used throughout this work is provided.

The first contributions of this thesis are presented in Chapter 3. In particular, the chapter starts by pointing out the lack of semantically rich resources for domain-specific and domain-agnostic applications using KGs. The first contribution targets the lack of publicly available resources in the educational domain for performing intelligent curriculum recommendations. More specifically, EducOnto and EduKG are thoroughly presented as adequate resources for conducting research at the intersection of recommender systems and explainable artificial intelligence in education. Then, the scope of Chapter 3 is voluntarily broadened. We first highlight the need for a larger and more diverse collection of benchmarks for KG-based applications, and highlight the relative paucity of schema-augmented datasets for KG-based applications. These issues led us to release PyGraft, an open-source Python tool for generating synthetic ontologies and KGs, endowed with an extensive set of parameters the user can control.

Providing a tool for generating datasets underpinned by a rich schema is expected to facilitate the development of neuro-symbolic approaches. The wealth of additional, semantically rich knowledge contained in such schemas can thus be leveraged to improve the accuracy of KGEMs for the LP task. However, in Chapter 4, we raise essential concerns about their evaluation. In particular, even when they would utilize additional information from a schema, KGEMs are solely evaluated with respect to rank-based metrics. Drawing on the inherent limits of this family of metrics, we also provide concrete arguments on why a multi-faceted evaluation procedure matters. We consequently contribute to this endeavour by proposing Sem@K, a novel semantic-oriented metric that focuses on the compliance of predictions with relation's domain and range. Sem@K comes in various flavors based on the nature of the dataset. We subsequently reassess popular KGEMs according to their semantic awareness, as measured by Sem$K$. This comprehensive study yields numerous insights into these models, with common trends noticeable at the level of families of models. In particular, the analysis shows that rank-based metrics' results do not necessarily correlate with the semantic awareness of KGEMs. This calls for increased cautiousness when using these models in real-world applications. We ultimately provide a concrete example by comparing the accuracy and semantic validity of different approaches in the context of recommender systems.

Chapter 5 goes a step beyond the mere evaluation of the semantic awareness of KGEMs. Now that we have presented a way (and a metric) to evaluate this dimension, we study how we can improve the semantic awareness of these models in the context of LP. This leads us to design semantic-enhanced models that, in contrast with agnostic models, leverage schema-based information in their training procedure. More specifically, we propose a way of incorporating background knowledge about relations' domain and range into popular loss functions for LP. The approach we propose substantially improves the semantic awareness of the studied models for LP, as evidenced by the notable increase in Sem@K values. In addition, in most cases we also observe a performance boost w.r.t. rank-based metrics. This contribution highlights

the importance of neuro-symbolic approaches. Building on these promising results, we then take a different outlook on our general intent: instead of enriching loss functions, we opt for mining knowledge about classes and subclass inheritances to avoid the random initialization of embeddings and, instead, initilize them according to the schema. However simple this approach is, it is demonstrated to increase the semantic awareness of KGEMs – which was our initial intent. Notably, the resulting embeddings are arguably more aligned with the overall semantics of the graphs, as evidenced by their versatility and reusability potential in other tasks such as node classification and entity clustering. This novel result – in addition to highlighting the benefits of neuro-symbolic approaches – opens a lot of potential research directions around the very notion of embeddings.

Finally, Chapter 6 summarizes the main contributions of this thesis and opens up future research perspectives.

Chapter 2

Link prediction in knowledge graphs

Contents

2.1 Knowledge graphs

2.1.1 Historical Perspective

The term *knowledge graph* itself was initially coined by Schneider in 1973 [148]. Schneider's work at that time was rooted in the field of computerized instructional systems for education. This is why, in this context, a KG referred to a graph where the nodes are units of knowledge (concepts) that students were expected to acquire, while edges denoted dependencies between knowledge units. In 1990, De Raedt *et al.* [36] propose a KG as a directed graph composed of a taxonomy of instances being connected with weighted edges to a taxonomy of classes. In 2005, Helms and Buijsrogge [66] propose a KG to model how knowledge is shared within an organisation, with nodes denoting stakeholders of different kinds and edges denoting certain types of knowledge from one actor to another.

Other phrases were used to represent similar notions by other authors, including "conceptual graphs' [153], "information graphs" [98], "information networks" [155], and "semantic networks" [119]. However, the proliferation of distinct terminologies was to gradually diminish

following the announcement of the Google Knowledge Graph [151] in 2012. With this announcement, Google primarily emphasized the concept of KG and outlined the potential applications it would enable. In this context, a KG was defined as "*[a graph] that understands real-world entities and their relationships to one another*" [151]. References to *knowledge graph* rapidly proliferated in the research literature. As Bergman pointed out in [9], Google's announcement marked a pivotal moment in the almost unanimous adoption of the term *knowledge graph*. However, Google did not provide any technical specification of what a KG is. Consequently, a formal definition was still lacking [44, 17]. As KGs were garnering increasing attention in academic literature, the need for formal definitions became imperative to clearly delineate what they actually constitute.

2.1.2 Definition

More than a decade has passed since Google's announcement. While the term *knowledge graph* has become widely accepted since then, its formal definition still lacks consensus in the research community. The term *knowledge graph* encompasses multiple interpretations; its definition is not unequivocal.

From a broad perspective, Hogan *et al.* [69] distinguish four categories of definitions. As the last category of definition bypasses the issue of actually defining what a KG is, we restrict ourselves to presenting the first three ones:

Category 1: a KG is defined as a graph in which nodes represent entities, and edges represent relationships between those entities. This definition often assumes a directed edge-labeled graph, or equivalently, a set of triples. As this definition is straightforward, intentionally permissive while still providing a framework sufficient enough to represent the data structure upon which embeddings would operate, it has been used in many works revolving around knowledge graph embeddings [18, 181, 164]. However, with this category of definitions, it is unclear whether and how *knowledge* is considered.

Category 2: A second prevalent definition states that a KG is a graph-structured knowledge base. The earliest documented usage of this definition in academia can be traced back to Nickel *et al.* [123] in 2016. However, the term *knowledge base* gained popularity in the context of rule-based expert systems [24] and was later utilized in the context of ontologies and other logical formalisms [21]. As such, this definition raises a follow-up question: Under this definition, can one have a KG without incorporating a logical formalism?

Category 3: The third category of definitions outline additional requirements a KG should comply with. Below, we list some relevant definitions falling under this category:

Ehrlinger and Wöß [44] contend that "*A knowledge graph acquires and integrates information into an ontology and applies a reasoner to derive new knowledge*". Therefore, they clearly differentiate the concepts of KG and ontology – the latter being considered synonymous with a knowledge base.

Paulheim [129] lists 4 criteria that characterize what a KG is. In particular, a KG "*mainly describes real world entities and their interrelations, organized in a graph; defines possible classes and relations of entities in a schema; allows for potentially interrelating arbitrary entities with each other; covers various topical domains*". Interestingly, Hogan *et al.* [69] point out that this definition rules out ontologies without individuals, graphs composed of word meanings (*e.g.* WordNet [110]) for not complying with the first two requirements, and domain-specific graphs (*e.g.* ClaimsKG [160], FoodKG [65], and K12EduKG [30]) for not complying with the last criterion.

Bellomarini *et al.* [8] define a KG as "*a semi-structured data model characterized by three components: (i) a ground extensional component, that is, a set of relational constructs for schema and data (which can be effectively modeled as graphs or generalizations thereof); (ii) an intensional component, that is, a set of inference rules over the constructs of the ground extensional component; (iii) a derived extensional component that can be produced as the result of the application of the inference rules over the ground extensional component (with the so-called "reasoning" process).*". To them, a KG is then a knowledge base extended with reasoning capabilities, which resembles the definition provided by Ehrlinger and Wöß [44].

In this manuscript, we use the inclusive definition proposed by Hogan *et al.* [69]:

Definition 2 (Knowledge Graph) *A knowledge graph is a graph of data intended to accumulate and convey knowledge of the real world, whose nodes represent entities of interest and whose edges represent relations between these entities.*

In this definition, the data graph conforms to a graph-based data model, such as a directed edge-labelled graph or a property graph. According to Hogan *et al.*, with this conceptual viewpoint, knowledge may be composed of simple statements, such as "Paris is the capital of France", or quantified statements, such as "all dogs are mammals". In the rest of this manuscript, we stick to the above definition that we clarify slightly in order to fit both our conceptual and experimental needs (Chapter 3 onward). In particular, in our experiments we clearly distinguish the schema-related or ontological triples (*e.g.* `Dogs subClassOf Mammals`) from the relational triples (*e.g.* `Paris capitalOf France`). In other words, we want to clearly delineate the *Terminological box* (*T-Box*) which defines class and property axioms, from the *Assertion box* (*A-Box*) which defines relations on individuals [69]. The above mentioned definition provided by Hogan *et al.* still suggests that a KG can be enriched with a schema or an ontology, *i.e.* a KG can model the full A-Box while also modelling a part of the T-Box. This is why, in the following, we unambiguously refer to a *schema* or *ontology* as modelling the *T-Box*, while *knowledge graph* is restricted to modelling the *A-Box*.

2.1.3 Different world assumptions and their consequences

In the context of knowledge representation and reasoning with KGs, the concept of the *world assumption* plays a pivotal role. The *world assumption* refers to the interpretative framework used to understand the presence or absence of links (relations) in a KG. Predominantly, the Closed World Assumption (CWA) is opposed to the Open World Assumption (OWA), while the Local Closed World Assumption (LCWA) appears as a variant that combines aspects of both CWA and OWA.

- **CWA** posits that what is not known to be true is necessarily false. In the context of KGs, this means that any relation not present in the graph is treated as non-existent or false. The CWA is commonly used in database systems and in many predictive approaches relying on KGs (*e.g.* link prediction, see Chapter 2).
- **OWA** assumes that the absence of a given fact in the KG does not imply its falsehood. The OWA is typically used in the Semantic Web and ontology-based systems.
- **LCWA** is a variant that combines aspects of both CWA and OWA. LCWA allows certain parts of the knowledge base to be treated under CWA while the rest under OWA. This can be useful in scenarios where some information domains are complete and others are not. The LCWA will be further discussed in Section 2.3.3.

The notion of *world assumption* will be further discussed in Chapter 2, more specifically in the context of negative sampling.

2.2 Popular knowledge graphs

As mentioned in Section 2.1, the conceptual boundary separating ontologies from KGs is permeable. In what follows, we temporarily loosen the definitions we agreed on in the previous section, and give examples of ontologies and KGs that are prominent and used in this thesis work – regardless of how they are defined by their creators and what sort of knowledge they encompass.

From a broad viewpoint, KGs can either cover multiple domains or be domain-specific. In this thesis work, we focus our attention on the most popular, open, and cross-domain (*i.e.* generic) KGs:

DBpedia

DBpedia [5] was originally a community-based project developed to extract structured information from Wikipedia. Although Wikipedia pages consist mostly of free text, they include structured information embedded in key-value pairs in Wikipedia "infoboxes" (*i.e.* the pull-out panels showing up in the top of many Wikipedia articles), categorization information, disambiguation pages, external links, geo-coordinates, images, redirects, etc. Specific extractors are designed to process these different kinds of information sources. After all this structured information is extracted, it is stored in a dump dataset that is further enriched by linking to external resources such as DrugBank [176], GeoNames[4], and WordNet [110]. The resulting, augmented content forms the DBpedia KG as we know it. Such KG can be queried using the SPARQL service endpoint identified by `http://dbpedia.org/sparql`, which handles more than 7.2 million queries daily on average. In DBpedia, entities are classified according to several schemata, each allowing to cover specific requirements [12]. These are, for example, the Simple Knowledge Organization System (SKOS) which is used to represent Wikipedia categories, Yet Another Great Ontology (YAGO) classification schema, and the own DBpedia ontology composed of classes such as `dbo:Country`, `dbo:Film`, and `dbo:Music`. The most up-to-date statistics[5] report that the DBpedia ontology covers 768 classes that are described by more than $3,000$ different properties, with about $4,233,000$ entities. The most recent snapshot releases describing these entities and how they interacts contain more than 850 million triples[6].

Freebase

Different from DBpedia whose content mainly derives from Wikipedia pages, Freebase's curation relied on the efforts of human editors. This way, the wide spectrum of human knowledge contained in Freebase (as, similar to DBpedia, Freebase aimed to be general-purpose and covers domains such as arts, politics, and sport) was expected to avoid the need for large-scale information integration processes associated with the decentralized nature of the SW. Importantly, Freebase relied on a lightweight typing system [15] which, instead of ensuring strict ontological compliance or logical consistency, implemented a loose set of specifications (*e.g.* datatypes, properties, schema definitions) which ultimately allowed for incompatible typing and property associations. Notably, after Freebase was acquired by Google in 2010, most of its content was

[4] `https://www.geonames.org/`
[5] `https://www.dbpedia.org/resources/ontology/`
[6] `https://www.dbpedia.org/blog/recap-2023-a-year-with-dbpedia/`

fed into the Google KG [151]. Freebase was shut down on May 2nd, 2016[7]. Its latest version contains approximately 50 million entities and 3 billion triples. Freebase schema encompasses around 27,000 entity types and 38,000 relation types [129]. Most of Freebase was then integrated into Wikidata [159], which we describe next.

Wikidata

Wikidata curation mechanism differs from DBpedia – which is manually curated and updated across different Wikipedia pages and languages. This leads to important data quality issues: for instance, if a tennis player was to win a Grand Slam, this achievement would need to be added to his records. Statistics about the Grand Slam would need to be updated, and more generally any tennis face-off in this Grand Slam would incur manual updates in the records of all the players who were part of it. Such manual curation can lead to incomplete or even contradictory information. However, it should be noted that Wikidata is not only a KG, but it is also a project led by the Wikimedia Foundation, along with other projects such as Wikipedia and Wikimedia Commons. This means that Wikidata is backed by dedicated staff members who are responsible for building and maintaining the infrastructure that relies on numerous patterns. One of these patterns is that, in Wikidata, adding a new fact triggers the automatic update of related resources across different languages [167]. In addition, with the sole exception of initial (and thereby unrelated) facts that can be added without reference [136], it is necessary to reference primary sources (*e.g.* adding provenance metadata). For example, this could be the geographic location, dates, and total prize of a tennis Grand Slam. Wikidata allows for collaboratively editing the data level but also some ontological assertions such as class inheritance and subproperties [135]. As a KG, Wikidata is garnering increasing popularity and its content is growing at a fast, steady pace since 2018 [45] It is now recognized as the largest public and cross-domain KG: more than 1.4 billion facts spanning over 100 million concepts are referenced, with contributions from approximately 560,000 editors [168]. Wikidata's outreach goes beyong the mere research community, as its content is directly being used in smart assistants such as Alexa or Siri [168]. In Wikidata, an item consists of an identifier, a label, and a description. Additional attributes such as aliases and statements are optional and are intended to enrich the information about a Wikidata entry.

YAGO

Similar to DBpedia, YAGO (standing for Yet Another Great Ontology) was built by extracting structured data from Wikipedia. These data are then mapped to the hierarchical structure of the the lexical resource WordNet [110] to generate a "light-weight and extensible ontology with high quality and coverage" [154]. This high quality and coverage was possible by extracting data from Wikipedia infoboxes and category pages – in a similar fashion as for DBpedia. YAGO was released in successive versions. YAGO 3 is probably still the most popular version so far. This version stands out in numerous ways: YAGO 3 encompasses entities and facts from 10 Wikipedias in different languages. It fuses WordNet taxonomy with Wikipedia category system to account for more than 350,000 distinct classes that can be attributed to entities. In addition, many of YAGO 3 facts and entities are enriched with spatial and temporal information. In total, YAGO 3 contains more than 16 million entities and 100 million triples connecting them [141]. To reflect the growing popularity of Wikidata, the newest YAGO 4 version is now based on it. YAGO 4 has the particularity that top-level classes are inherited from `schema.org` and `bioschemas.org`, while

[7] `https://en.wikipedia.org/wiki/Freebase_(database)`

the lower-level classes are composed of Wikidata classes. Therefore, YAGO 4 can be considered as a refined version of Wikidata, containing more than 50 million entities and 2 billion facts [131].

2.3 Link prediction with knowledge graph embedding models

2.3.1 Overview

While there is a vast quantity of relational data publicly available, KGs remain inherently incomplete due to several factors, which stem from their construction process, the assumptions about the world that are made, and the nature of the information they aim to represent. Overall, a non-exhaustive list of common reasons explaining their incompleteness is the following:

- *Limitations in Automated Information Extraction*: Automated processes for extracting information from texts and populating KGs, such as natural language processing (NLP) algorithms, are not perfect. They may miss or misinterpret information, leading to incomplete data in the graph. For instance, NELL [112] is originally built and continuously updated on the basis of information extraction methods operating on crawled web pages. This inevitably leads to incorrect statements, *e.g.* "Nepal is a country also known as United States"[8].

- *Selective Coverage Based on Interests and Priorities*: The focus of a KG may be influenced by the interests and priorities of its creators and curators. This can lead to selective coverage of topics, with some areas being well-represented while others being neglected.

- *Human Error in Data Curation*: Manual curation and data entry processes are prone to human error, which can result in missing or incorrect information in the knowledge graph.

- *Dynamic and Evolving Information*: Information changes over time. As information about the world evolve, new facts need to be added and existing facts may need to be updated or even invalidated.

Other reasons could lead to KG incompleteness, *e.g.* resource and technical limitations, data integration challenges, and privacy and ethical considerations. Regardless of the actual reason(s) for a KG being incomplete, *knowledge graph refinement* [129] is concerned with completing or correcting information found in KGs. Using refinement methods necessarily implies a trade-off between coverage and correctness when trying to infer, add, update, or remove pieces of information.

In Freebase, over 70% of people have no known place of birth, and 99% have no known ethnicity [16]. In response to this information shortage, *Knowledge graph completion* (KGC) is concerned with enriching an existing KG by adding new entities, relationships, or both. Therefore, KGC can be seen as a subset of knowledge graph refinement, where the emphasis is put on expanding the knowledge covered by KGs. The focus of KGC is not unique, as it can involve identifying and integrating new entities that are not currently represented in the graph, inferring and adding new relationships (edges) between existing entities or between new and existing entities, or even leveraging external data sources to fill in the gaps in KGs. The intended purpose of completing KGs is to make them inherently more useful in applications and downstream tasks. For instance, a more comprehensive KG is expected to foster better accuracy in question-answering systems, and lends itself better to the recommendation of relevant items to end-users.

[8] `https://en.wikipedia.org/wiki/Never-Ending_Language_Learning#Reception`

Link prediction (LP) is itself a subset of KGC that specifically focuses on predicting whether a link (relationship) should exist between two entities in a KG. The commonly shared understanding of the LP task is that, based on a given entity and an existing relation, the objective is to predict the missing entity. More formally, LP is a task that consists in exploiting the existing facts in a KG to infer missing ones. This amounts to predicting the ground-truth entity for $(?, r, t)$ (head prediction) and $(h, r, ?)$ (tail prediction). The entity to predict will be referred to as the *target* entity, while the known entity will be denoted as the *source* entity [144].

LP should be distinguished from *relation prediction*, which focuses on predicting the relation that holds between two existing entities in the KG, and *triple classification*, which deals with classifying (complete) facts as true or false. In this thesis, we exclusively focus on LP.

More specifically, we restrict our discussion to LP in the *transductive* setting, *i.e.* all the candidate entities in the triples to predict are already observed in the KG. This differs from the *inductive* setting, where new entities or relations can emerge at testing time. A similar situation often appears in *zero-shot* LP. For more details about these tasks, the reader is advised to refer to [73].

Ultimately, the task of LP can be tackled with different approaches, *e.g.* path-based, rule-based, and embedding-based models [32].

- **Path-based methods** exploit the paths connecting entities in a KG, under the assumption that these paths can be used to reveal potential relationships between entities [183]. They do so by leveraging different statistical features of nodes (entities) and edges (relations), *e.g.* the outgoing degree and incoming degree of nodes and the adjacency matrix.

- **Rule-based methods** incorporate prior knowledge and structured information to automatically mine logical rules in a KG so that new knowledge (facts) can be inferred. These rules are typically expressed in some form of logical or symbolic language, such as first-order logic or Prolog. Extracted rules can take various forms, including Horn clauses, association rules, or more complex logical expressions. Each rule may be associated with a confidence score that reflects the strength or reliability of the rule. Rule-based models are often used where explicit domain knowledge or logical constraints are essential for making accurate predictions. They can also be used when interpretability and explainability are critical, as predictions can be traced back to specific rules. However, owing to the exponential search space, rule-based models do not scale well with large KGs [99].

- **Embedding-based models** are the focus of this thesis. Compared to rule-based models, they can scale to large KGs and often outperform rule-based methods in terms of prediction accuracy. They can be used effectively even when little prior domain-specific knowledge is available, and they have been proven to still capture complex patterns and semantic relationships. They have consequently been garnering increasing attention in the research community. Overall, embedding-based models are ML techniques that rely on vector representations of entities and relations, a.k.a *embeddings*. This whole chapter is dedicated to explaining their intrinsic characteristics.

With the expansion of computing resources and accompanying machine and deep learning methods, the task of LP in KGs has garnered increasing interest. In particular, embedding-based models have emerged as the favored approach for completing these KGs and are the focus of this chapter, which discusses ML models relying on vector representations of entities and relations, *i.e. knowledge graph embedding models* (KGEMs). Inspired by Word2Vec [109], their main objective is to generate a continuous, low-dimensional vector representation of the KG

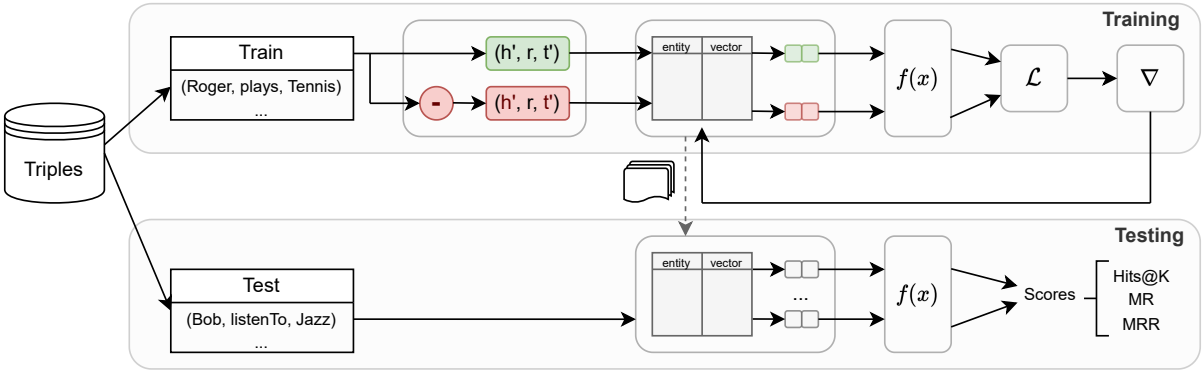


Figure 2.1: Overview of the pipeline for training and testing KGEMs for link prediction

entities and relations, dubbed as *knowledge graph embeddings* (KGEs). Then these numerical representations are directly used to infer new facts. By "low-dimensional", it is frequently assumed that the dimension d for representing entities and relations ranges between 50 and 500 [144], with some models requiring higher values for d. The resulting embeddings can be further utilized for downstream applications such as recommender systems.

The general pipeline for training and testing KGEMs is depicted in Fig. 2.1. It is made of several components that interact in a sequential and iterative manner. Each component plays a crucial role in understanding and working with KGEMs. Consequently, they are individually discussed in greater depth in Sections 2.3.2 (interaction function), 2.3.3 (negative sampling), 2.3.4 (training), and 2.3.5 (evaluation).

Before diving into each of these components, in the following we detail some conceptual as well as technical topics about KGEMs. This preliminary discussion aims at providing a general overview of the field, before touching more complex aspects.

Batching procedure. The aforementioned KGEMs involve training, validation, and testing phases for effective learning and evaluation. It implies that KG datasets are split into training, validation and testing batches of triples. As this thesis focuses on the transductive setting, these batches of triples cannot be formed randomly. In particular, we need to ensure that each entity and relation in the validation (resp. testing) set also appears in the training set. Otherwise, the embeddings of some entities and/or relations are not learnt on the training data, which boils down to performing inductive or zero-shot LP [73].

Model classification. KGEMs are traditionally divided in several families depending on such aspects as their representation space [28], the form of their interaction function (a.k.a scoring function) [89], or their encoding strategy [89]. KGEMs are commonly classified as belonging to one of the following families: geometric, multiplicative, and deep models (Fig. 2.2). While representatives of each family are presented in greater depth in Section 2.4, we still present the main characteristics that unify models into such families:

- **Geometric models** interpret relations as geometric transformations in the embedding space and rely on a distance-based function to assess the plausibility of a fact. Most of them are further classified as translational models; they model relations as translations between two entities. For a given fact (h, r, t), they seek to satisfy the equation $h + r \sim t$. Well-known representative models of this type are TransE [18], TransH [174], and TransD [88] (colloquially referred to as TransX models as they all are extensions of the seminal TransE

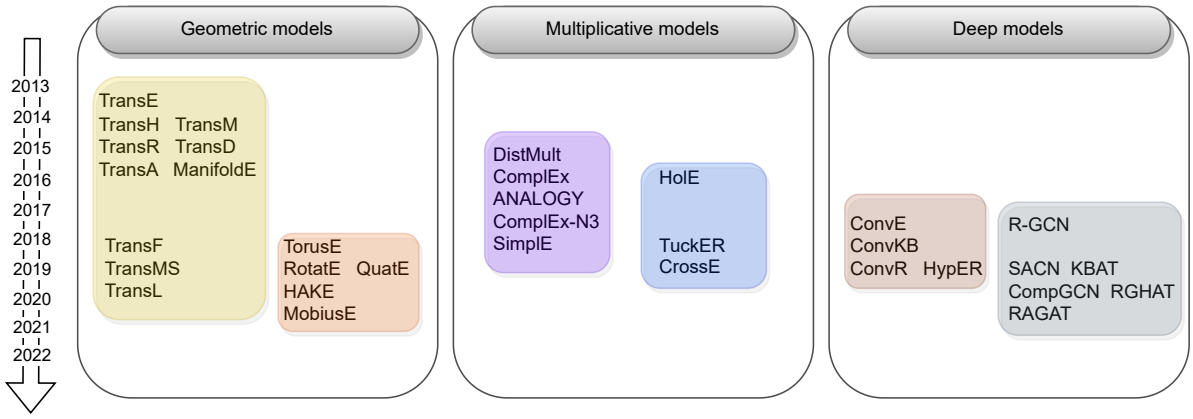


Figure 2.2: Taxonomy of KGEMs with a timeline

model). More expressive KGEMs include operations that go beyond pure translations, *e.g.* rotations. Such roto-translational models perform transformations based on rotations, either in combination or by replacing pure translations. RotatE [157], MobiusE [31], and TorusE [42] fall under this hood.

- **Multiplicative models** assess the plausibility of a fact using a multiplicative scoring function. Whereas the previously described geometric models are inherently associated to the notion of distance between elements, multiplicative models are best seen as measuring the similarity between them. Multiplicative models are further subdivided into bilinear and non-bilinear models. Bilinear models such as RESCAL [122], DistMult [122] and ComplEx [164] represent entities as vectors and relations as matrices. The interaction between entities and relations is modeled by multiplying the entity vector with the relation matrix. Non-bilinear models such as combine the head, relation and tail embeddings using formulations that differ from the strictly bilinear product. For example, HolE [121] uses the cyclic correlation of vectors to represent entity pairs and TuckER [7] relies on the Tucker (tensor) decomposition.

- **Deep models** leverage neural network architectures to learn representations of KG entities and relations. These models are predominantly classified into convolutional-based and graph-based approaches. Convolutional-based models utilize convolutional neural networks (CNNs) to extract features from the embeddings of entities and relations. They apply convolutional operations over embeddings to capture local and global patterns. Notable examples include ConvE [39] and ConvKB [120]. Graph-based models exploit graph neural networks (GNNs) to learn embeddings by considering the graph structure of the KG. These models use the connectivity patterns of entities and relations to enrich the representation of each node (entity) in the graph. Examples include R-GCN [147], SACN [149], KBGAT [118], and CompGCN [166].

Representation space. Although KGEMs tend to be primarily differentiated according to their encoding strategy and interaction function (see the three families described above), another classification based on their representation space exists. To this respect, KGEMs either rely on pointwise, complex, manifold, or gaussian representation spaces [89]. These are briefly summarized below:

- **Pointwise** representation space implies that entities and relations are represented as points in an Euclidean space, while common operations include vector addition and subtraction. Therefore, it is straightfoward to understand and visualize the resulting embedding space, making it a popular choice in early KGEMs (*e.g.* TransE). However, KGEMs that belong to this family may struggle with complex relation types like many-to-many relations due to the Euclidean nature of the embedding space.
- **Complex** representation space denotes the choice of complex numbers (real and imaginary components) to represent KG entities and relations. It offers more degrees of freedom compared to real-valued embeddings, enabling better modeling of asymmetric relations (*e.g.* with ComplEx [164]). However, the richer representations come at some cost: they can be more challenging to interpret and may require more parameters than real-valued models.
- **Manifold** representation space involves representing entities and relations on non-Euclidean manifolds, such as spherical or hyperbolic spaces (*e.g.* TorusE [42]). Opting for this representation space is particularly effective for modelling hierarchical data. It also offers some flexibility to capture complex structures not easily represented in Euclidean space. However, it requires specialized mathematical tools and understanding, which can add to the complexity of model development and analysis.
- **Gaussian** representation space models entities and relations as Gaussian distributions rather than points. It enables capture the uncertainty and variability of entity and relation representations, and thereby allows for a probabilistic interpretation of the embeddings (*e.g.* with TransG [178]). This comes at a potentially increased computational complexity due to the need to handle distributions instead of point estimates.

Model expressiveness. Regardless of the family of models considered (although it can play a part, too), KGEMs expose different modeling abilities on different types of relational patterns. One of the reasons that frequently led researchers to propose new relational models was to address the shortcomings of previous models in terms of representation expressiveness. For instance, TransE does not properly handle 1-to-N, N-to-1, nor N-to-N relations [171]. Many of the KGEMs that were subsequently introduced aimed at alleviating the shortcomings of TransE regarding these cardinality restrictions. Moreover, relations can be further expressed w.r.t. other existing relations, or can pose constraints between subject and object entities. In the following, we elaborate on the main properties and link them to their expression in terms of `OWL`/`RDFS` as detailed in Section **??**.

Definition 3 *A relation r is defined as symmetric (resp. asymmetric) if* $\forall x, y$

$$r(x, y) \Rightarrow r(y, x) \quad (resp. \quad r(x, y) \Rightarrow \neg r(y, x))$$

In `OWL`, a symmetric (resp. asymmetric) relation can be expressed with `owl:SymmetricProperty` (resp. `owl:AsymmetricProperty`).

Definition 4 *A relation r_1 is inverse to a relation r_2 if* $\forall x, y$

$$r_2(x, y) \Rightarrow r_1(y, x)$$

In `OWL`, such relation can be expressed with `owl:inverseOf`.

Definition 5 *A relation r_1 is composed of relations r_2 and r_3 if* $\forall x, y, z$

$$r_2(x, y) \wedge r_3(y, z) \Rightarrow r_1(x, z)$$

Composition is not directly modelled in `OWL` and `RDFS`. However, it can be represented in `OWL` using property chains to generate hand-crafted rules.

Model	Symmetry	Asymmetry	Inversion	Composition
TransE	✗	✓	✓	✓
TransR	✗	✓	✓	✓
DistMult	✓	✗	✗	✗
ComplEx	✓	✓	✓	✗
RotatE	✓	✓	✓	✓

Table 2.1: Relational pattern modelling of several KGEMs

Table 2.1 summarizes the modelling capabilities of several mainstream KGEMs over the aforementioned relational patterns. Notably, it has been recently demonstrated that the theoretical support of KGEMs for given relational patterns does not necessarily lead to better performance on triples exhibiting such relational patterns [91].

2.3.2 Interaction function

Completing a KG is often cast as a supervised, binary classification task: classifying triples as either true or false. This way, training KGEMs does not differ much compared to traditional ML approaches that rely on discriminating positive from negative samples. However, as mentioned in Section 2.1, KGs are not expected to store negative facts. Therefore, the training data typically includes examples from only one class label (*i.e.* positive facts). This situation poses a fundamental challenge for training KGEMs. Details on how to generate negative statements are more thoroughly discussed in Section 2.3.3. For now, we assume the availability of both true and false triples.

A common approach to circumvent the issue of the sole availability of positive statements involves approximating a theoretical model that assesses the plausibility of a given triple, *i.e.* its probability of being true. Consequently, models are conceptualized as parametric interaction (or scoring) functions dedicated to estimating this likelihood (Definition 6).

Definition 6 (Interaction function) *Given a knowledge graph $\mathcal{K} = (\mathcal{E}, \mathcal{R}, \mathcal{T})$, where $\mathcal{E}$ represents the set of entities, $\mathcal{R}$ the set of relations, and $\mathcal{T}$ the set of relational triples, an interaction function is defined as a mapping function $f : \mathcal{E} \times \mathcal{R} \times \mathcal{E} \to \mathbb{R}$ that allocates real-valued scores to triples (h, r, t), with $h, t \in \mathcal{E}$ and $r \in \mathcal{R}$. The design of this function should ensure higher scores for triples that represent true facts and lower scores for false triples.*

This approach for assigning a plausibility score to triples is based on the premise that a model that is capable of correctly assessing the truthfulness of a triple should have encapsulated the essential information about the entities and the relation within its internal representation.

Interaction functions are by far the most important component for understanding and defining the nature of KGEMs, as evidenced by the most frequent classification of KGEMs based on their interaction functions (see Section 2.3). However, as depicted in Fig. 2.1, the interaction function (denoted as $f(x)$ in the schematic representation) is one of the many characteristics that

need to be defined – along with the negative sampling strategy (Section 2.3.3), the chosen loss function and optimization procedure (Section 2.3.4), and the metrics to evaluate the resulting model (Section 2.3.5). Recent works highlight that the choice of the interaction function is not the unique criterion leading to good performance; the choice of loss function and the training approach are also crucial [3]. This is why we discuss these essential components in the following sections.

2.3.3 Negative sampling

As previously discussed in Section 2.3.2, training a KGEM to predict the gold entity in incomplete triples requires it to position entities and relations in the embedding space in such a way that its interaction function can effectively assign a higher plausibility score to the ground-truth than to non ground-truth triples. In particular, when predicting the missing entity for $(?, r, t)$ (head prediction) or $(h, r, ?)$ (tail prediction), every candidate entity e that leads to (e, r, t) (resp. (h, r, e)) being a positive fact, thereby leads to a positive sample. As most KGs do not come with predefined negative facts, we need to generate negative triples to guide the model in learning good representations for KG entities and relations by successfully discriminating positive facts from negative ones. A common approach is to corrupt positive facts to generate negative counterparts. This process called *negative sampling* (NS) is at the core of KGEM training. Different world assumptions and sampling procedures underpin the main NS strategies found in the literature. We consequently discuss these aspects in the following.

World assumption

As stated in Section 2.1.3, different world assumptions can prevail when representing knowledge and reasoning with it. In this section, we revisit the notion of world assumption in the context of performing LP with KGEMs. In such context; the CWA usually prevails. However, a finer-grained distinction needs to be made. In the context of negative sampling, the OWA and LCWA that we defined in Section 2.1.3 have specific implications [3]. Additionally, we introduce the notion of stochastic local closed world assumption (SLCWA). Importantly, the choice of world assumption does not affect the evaluation protocol (Section 2.3.5). However, it does affect the space of potential negative samples. In the following, we note $\mathcal{T}(h, r)$ (resp. $\mathcal{T}(r, t)$) as the set of triples obtained by replacing the tail t (resp. head h) of a positive triple successively by all the other observed entities in the KG (Equations 2.1 and 2.2):

$$\mathcal{T}(h, r) = \{(h, r, t') \mid t' \in \mathcal{E} \wedge t' \neq t\} \tag{2.1}$$

$$\mathcal{T}(r, t) = \{(h', r, t) \mid h' \in \mathcal{E} \wedge h' \neq h\} \tag{2.2}$$

We can thus define the set of candidate negative facts (Equation 2.3):

$$\mathcal{N} = \bigcup_{(h,r,t) \in \mathcal{K}} \mathcal{T}(h, r) \cup \mathcal{T}(r, t) \tag{2.3}$$

Importantly, the resulting sets of triples – $\mathcal{T}(h, r)$, $\mathcal{T}(r, t)$, and $\mathcal{N}$ – may contain *false negative* triples, *i.e.* triples that are observed in the KG and labelled as positive triples. Ideally, we should therefore consider the following filtered sets:

$$\mathcal{T}^-(h,r) = \{(h,r,t') \mid (h,r,t') \in \mathcal{T}(h,r) \wedge (h,r,t') \notin \mathcal{K}\} \quad (2.4)$$

$$\mathcal{T}^-(r,t) = \{(h',r,t) \mid (h',r,t) \in \mathcal{T}(r,t) \wedge (h',r,t) \notin \mathcal{K}\} \quad (2.5)$$

$$\mathcal{N}^- = \bigcup_{(h,r,t)\in\mathcal{K}} \mathcal{T}^-(h,r) \cup \mathcal{T}^-(r,t) \quad (2.6)$$

OWA. In this setting, for any positive triple (h,r,t) in the KG, the following two sets of negative triples are constructed: $\mathcal{T}^-(h,r)$ and $\mathcal{T}^-(h,r)$. This choice leads to the complete set $\mathcal{N}^-$ of all the negative triples that can possibly be generated for a given ground-truth triple (h,r,t). However, in practice, the full OWA is barely used and most works either opt for the LCWA or the SLCWA.

LCWA. Compared to the OWA detailed above, the LCWA is oftentimes defined as building only $\mathcal{T}^-(h,r)$ as the set of negative triples, and training the KGEM to discriminate a positive triple (h,r,t) against all its negative counterparts found in $\mathcal{T}^-(h,r)$. The LCWA is used for training models such as ConvE [39] and Tucker [7] that both rely on 1-N scoring [39].

SLCWA. Instead of considering all the negative triples in $\mathcal{N}^-$, a predefined number of negative triples is sampled from this set. Notably, the first two NS strategies we elaborate on in what follows – Uniform NS (UNS) and Bernoulli NS (BNS) – are based on the SLCWA, and several KGEMs are originally trained using this approach. It should be noted that implementations might differ between works, and most of them actually sample in $\mathcal{N}$ instead of $\mathcal{N}^-$ (*e.g.* TransE [18]). However, in most cases $|\mathcal{N}| \gg |\mathcal{K}|$, such that the likelihood of sampling a false negative triple is low. Therefore, this choice of skipping the additional filtering step is sometimes chosen to reduce the computational cost and training time.

Sampling strategies

In the following, we detail the main NS strategies found in the literature.

Uniform NS (UNS). This NS strategy first proposed by Bordes *et al.* [18] follows the CWA and corrupts positive triples by replacing either the head or the tail by a randomly chosen entity from the KG, with uniform probability. If the resulting triple does not belong to the existing KG, it is considered to be a proper negative sample under this approach. Algorithm 1 describes the whole procedure.

UNS is used in many works (*e.g.* [105, 164]) due to its simplicity and computational efficiency. However, it can lead to negative triples too easy to discriminate. Therefore, at some point during the training phase, the model cannot further improve the quality of its KGEs due to trivial negative samples that are not useful for refining latent representations anymore. This issue known as the *vanishing gradient problem* has been extensively studied, and later works proposed more sophisticated negative samplers to generate higher-quality negative triples [27, 186].

Bernoulli NS (BNS). A first approach to improve the quality of corrupted triples is to maximize their probability of representing false facts in the first place. Due to the CWA that underpins Uniform NS [18], a corrupted triple is considered false only if it does not appear in the observed KG. Consequently, a KG that is moderately to highly incomplete would leave a lot of room for generating new, unobserved triples that actually represent true facts. To remedy this, Wang

Algorithm 1 Uniform Negative Sampling

Input: (h, r, t), a positive triple
Output: (h', r, t'), a negative triple
Data: $\mathcal{K}$, an existing KG
$(h', t') \leftarrow (h, t)$
while $(h', r, t') \in \mathcal{K}$ **do**
    $u \leftarrow x \sim U(0, 1)$
    if $u < \frac{1}{2}$ **then**
        $h' \leftarrow$ random entity
    else
        $t' \leftarrow$ random entity
return (h', r, t')

et al. [174] propose BNS. Although BNS still follows the CWA, it reduces the likelihood of accidentally sampling a true fact as a negative example by adjusting the sampling process based on the characteristics of the relations. More specifically, a Bernoulli parameter p^r is computed for each relation r as follows:

$$p^r = \frac{\rho^r_{t,h}}{\rho^r_{t,h} + \rho^r_{h,t}} \tag{2.7}$$

In the above formula, $\rho^r_{t,h}$ (resp. $\rho^r_{h,t}$) denotes the average number of tail entities per head entity (resp. head entities per tail entity) for a given relation r. The Bernoulli parameter p^r is the probability to replace the head entity for any triple involving r. Although BNS is still relatively straightforward and does not incur much computational overhead, it is still widely used in many recent works (*e.g.* [42, 31]). Algorithm 2 describes the whole procedure.

Algorithm 2 Bernoulli Negative Sampling

Input: (h, r, t), a positive triple
Input: p_r, Bernoulli parameter for relation r
Output: (h', r, t'), a negative triple
Data: $\mathcal{K}$, an existing KG
$(h', t') \leftarrow (h, t)$
while $(h', r, t') \in \mathcal{K}$ **do**
    $u \leftarrow x \sim U(0, 1)$
    if $u < p_r$ **then**
        $h' \leftarrow$ random entity
    else
        $t' \leftarrow$ random entity
return (h', r, t')

KBGAN. Cai *et al.* [27] introduced an adversarial learning framework to improve KGEs. Their negative sampler – called KBGAN – relies on the generative adversarial networks (GANs) that were previously introduced in [56]. Cai *et al.* use two different KGEMs: one KGEM acts as a negative sample generator, and assists the training of the other KGEM, which acts as the discriminator in the GAN framework. Notably, as a GAN-based approach, it alleviates

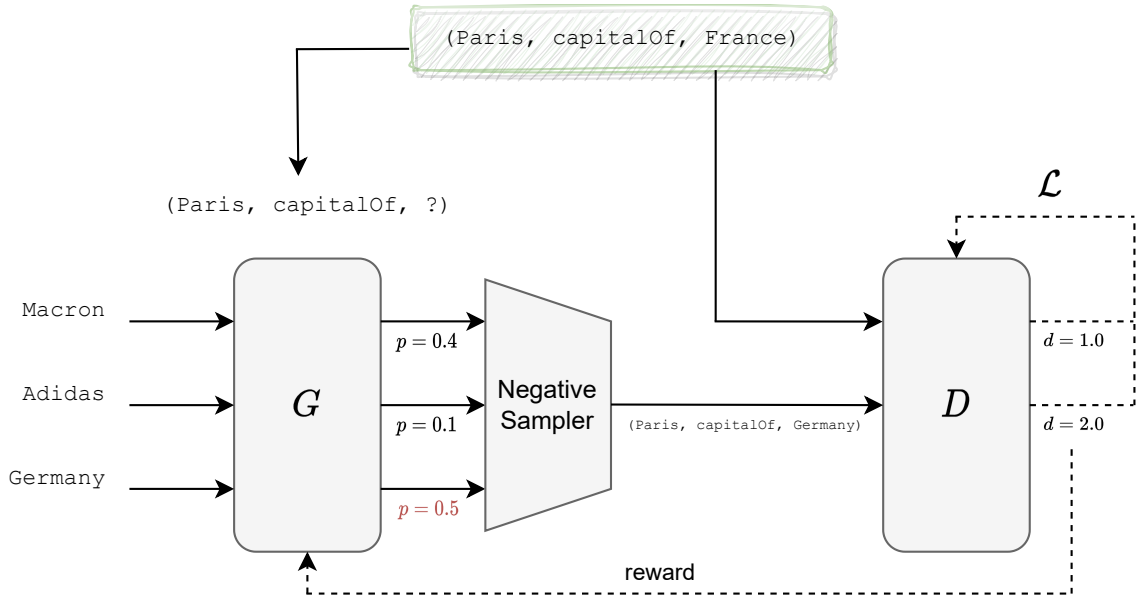


Figure 2.3: KBGAN overview. For a given positive triple `(Paris, capitalOf, France)`, several negative counterparts are generated by the generator. The likelihood or plausibility of each corrupted triple is determined probabilistically. The negative triple with the highest probability is chosen as the final negative triple to be discriminated against the positive triple `(Paris, capitalOf, France)` inside the discriminator. Finally, the marginal loss function is computed to update the discriminator, while the reward function is used to improve the quality of the generated triples.

the aforementioned vanishing gradient problem by only sampling negative triplets with large scores. In addition, the proposed approach is model-agnostic and can therefore use a broad array of embedding models as generator and discriminator. An overview of KBGAN is depicted in Fig. 2.3.

NSCaching. Zhang *et al.* [186] highlight the fact that GAN-based negative samplers such as KBGAN are harder to train and usually need reinforcement learning. Based on the observation that negative triples with large scores are scarce yet crucial to training KGEMs by effectively avoiding the vanishing gradient problem, they propose to keep track of such high-quality negative triples by caching them. Their method can be seen as a distilled version of previously introduced GAN-based samplers, as NSCaching does not aim to fully fit the underlying distribution of negative triples, but instead achieves a good balance between exploration and exploitation. Importantly, NSCaching addresses two main challenges inherent to NS: modeling the dynamic distribution of negative triples, and sampling them in an efficient manner. Similar to KBGAN, their approach is not tied to particular interaction functions, but can accommodate many existing KGEMs.

Structure Aware NS (SANS). Following the primary concern of NSCaching about proposing negative samplers with fewer parameters, Ahrabian *et al.* propose SANS [1], an efficient NS strategy that leverages the topology of a graph by selecting negative samples from the K-hop neighborhood of entities. They empirically demonstrate the importance of incorporating graph structure for NS, as SANS almost systematically generates harder negative triples than uniform random NS [19] and KBGAN [27] over the three datasets they used.

Simple NS (SNS). While SANS exploits the locality of an entity's neighborhood based on graph-based information (*i.e.* whether two entities can be linked through a path of a given length), Simple NS [84], in contrast, is based on proximity in the embedding space. In particular, the approach follows the assumption that the entities which are closer in the embedding space to the corrupted entity are able to provide high-quality negative triples. Islam *et al.* consequently propose a novel rule mining method directly operating in the embedding space. Doing so, they are also able to extract final rules and to explain the resulting link predictions.

The interested reader can find more negative samplers and a comprehensive analysis of their respective characteristics in [96]. Besides, a few recent works demonstrate that it is possible to dispense with negative sampling [103, 61].

2.3.4 Training

KGEM training implies finding optimal values for its entity and relation embeddings, so that the interaction function (Section 2.3.2) gives high scores to true facts and low scores to false ones. Typically, this is achieved by first generating a set of negative triples to discriminate from their positive counterparts (Section 2.3.3). The purpose of this section is to explain what happens after negative triples are generated and scored along with positive ones. At this point, we need to minimize a global measure of the errors made on known true triples compared their negative counterparts. In other words, there is a need to define a *loss function* whose computed values for each pair of positive/negative triples denotes the gap between the corresponding scores. Ideally, the loss should be high for close scores – implying that the model struggles to differentiate a positive sample from its negative counterpart, and should therefore differentiate their embeddings with greater emphasis. As the candidate loss functions for training KGEMs are chosen to be differentiable, they are optimized by performing gradient descent. This section addresses the aforementioned two aspect of KGEM training, *i.e.* loss functions and optimization.

Loss functions

Few works revolve around the influence of loss functions on KGEM performance [3, 115, 114]. Mohamed *et al.* [115] point out the lack of consideration regarding the impact of loss functions on KGEM performance. Experimental results provided in [3] indicate that no loss function consistently provides the best results, and that it is rather the combination between the scoring and loss functions that impacts KGEM performance. In particular, some scoring functions better match with specific loss functions. For instance, Ali *et al.* show that TransE can outperform state-of-the-art KGEMs when configured with an appropriate loss function. Likewise, Mohamed *et al.* [115] show that the choice of the loss function significantly influence KGEM performance. Consequently, they provide an extensive benchmark study of the main loss functions used in the literature. Namely, their analysis relies on a commonly accepted categorization between pointwise and pairwise loss functions. The main difference between pointwise and pairwise loss functions lies in the way the scoring function, the triples, and their respective labels are considered all together. Under the pointwise approach, the loss function relies on the predicted scores for triples and their actual label values, which is usually 1 for positive triple and 0 (or -1) for negative triples. In contrast, pairwise loss functions are defined in terms of differences between the predicted score of a true triple and the score of a negative counterpart.

The main loss functions used for performing LP are the pairwise hinge loss (PHL) [18], the 1-N binary cross-entropy loss (BCEL) [39], and the pointwise logistic loss (PLL) [164]. Their formulas are recalled in Equations 2.8, 2.9, and 2.10.

$$\mathcal{L}_{PHL} = \sum_{q \in \mathcal{T}^+} \sum_{q' \in \mathcal{T}^-} \left[\gamma + f(q') - f(q)\right]_+ \tag{2.8}$$

where $\mathcal{T}$, f, and $[x]_+$ denote a batch of triples, the scoring function, and the positive part of x, respectively. $\mathcal{T}$ is further split into a batch of positive triples $\mathcal{T}^+$ and a batch of negative triples $\mathcal{T}^-$. γ is a configurable margin hyperparameter specifying how much the scores of positive triples should be separated from the scores of corresponding negative triples.

$$\mathcal{L}_{BCEL} = -\frac{1}{|\mathcal{T}|} \sum_{q \in \mathcal{T}} \ell(q) \log(f(q)) + (1 - \ell(q)) \log(1 - f(q)) \tag{2.9}$$

where $\ell(q) \in \{0, 1\}$ denotes the true label of the triple q and $\mathcal{T}$ is a batch with all possible triples of the form $(h, r, *)$.

$$\mathcal{L}_{PLL} = \sum_{q \in \mathcal{T}} \log(1 + \exp^{-\ell(q) \cdot f(q)}) \tag{2.10}$$

where $\ell(q) \in \{-1, 1\}$ similarly denotes the true label of q.

Optimization

The choice of initialization method for setting parameters' values can significantly impact the convergence speed, final performance, and ability of the model to escape local minima. Most popular choices are random normal, random uniform, and Xavier [55] initializations. It is also possible to use pre-trained embeddings [109, 133] or rely on domain-specific initializations [134, 78]. However, these more sophisticated initialization strategies go beyond the scope of this thesis. Note that in the following, we refer to *model parameters* as a broader term than embeddings or KGEs. This is because a large number of KGEMs - especially deep models - maintain a set of parameters that includes but is not restricted to embeddings, and in general all these parameters are updated.

The core of optimization is to iteratively adjust model parameters, thereby minimizing the loss function. This is achieved through gradient descent (GD), a method where the gradient of the loss relative to the parameters of the model is computed. Subsequently, the parameters are updated in the direction opposite to that of the gradient (as stepping in the direction of the gradient would lead to a local maximum for the loss function).

There are three variants of GD, which differ in how much data are used to compute the gradient.

In vanilla GD, the gradient of the loss function is computed w.r.t. to the model parameters θ for the entire training dataset (Algorithm 3). Using vanilla GD can be impractical. In fact, computing the gradient over a large dataset is often computationally intensive and time-consuming, especially for large models. In addition, each iteration of GD (one epoch) requires processing the entire dataset. Provided that the datasets fits in memory, it means that updates to the model parameters happen at the end of each epoch. This can slow down the overall training process. Training with the full dataset at each step can also induce overfitting to the training data and poor generalization capabilities of the resulting model.

To remedy these issues, Stochastic Gradient Descent (SGD) trades the exact full gradient by an approximate proxy value computed for each training sample along with its label (for KGEM, this amounts to one training triple q with its label $\ell(q)$. Algorithm 4 details the procedure in the context of training KGEMs.

Algorithm 3 Vanilla Gradient Descent

Input: Learning rate α, initial parameters θ
Output: Optimized parameters θ
for each epoch **do**
    Compute the full gradient: $\nabla_\theta \mathcal{L}(\theta)$ ▷ (the exact calculation is loss-dependent)
    Update the parameters: $\theta \leftarrow \theta - \alpha \cdot \nabla_\theta \mathcal{L}(\theta)$
return θ

Algorithm 4 Stochastic Gradient Descent

Input: Learning rate α, initial parameters θ, triples $\mathcal{T}$
Output: Optimized parameters θ
for each epoch **do**
    Randomly shuffle the triples in $\mathcal{T}$
    for each triple q from $\mathcal{T}$ **do**
        Compute the gradient estimate: $\nabla_\theta \mathcal{L}(\theta, q, \ell(q))$
        Update the parameters: $\theta \leftarrow \theta - \alpha \cdot \nabla_\theta \mathcal{L}(\theta, q, \ell(q))$
return θ

In general, the gradient is rarely computed over the whole training set or one single data point (triple). In practice, mini-batch GD is more frequently used. It can be seen as an intermediate between vanilla GD and SGD, and consists in computing losses over mini-batches of triples of reasonable size (*e.g.* 128, 256, 512 [3]). Sometimes, the term SGD is also employed to refer to mini-batch GD. In this work, we clearly differentiate them. Mini-batch GD is more formally introduced in Algorithm 5.

Algorithm 5 Mini-batch Gradient Descent

Input: Learning rate α, initial parameters θ, triples $\mathcal{T}$
Output: Optimized parameters θ
for each epoch **do**
    Randomly shuffle the triples in $\mathcal{T}$
    for each (mini-batch $\mathcal{T}$ in $\mathcal{T}$ **do**
        Compute the gradient estimate: $\nabla_\theta \mathcal{L}(\theta, q^{(i:i+|\mathcal{T}|)}, \ell(q^{(i:i+|\mathcal{T}|)}))$
        Update the parameters: $\theta \leftarrow \theta - \alpha \cdot \nabla_\theta \mathcal{L}(\theta, q^{(i:i+|\mathcal{T}|)}, \ell(q^{(i:i+|\mathcal{T}|)}))$
return θ

Whether using vanilla GD, SGD, or mini-batch GD, common issues arise. First, choosing a proper learning rate is not straightforward; it would require many attempts and manual tweaking. Secondly, the learning rate should ideally be set at its maximum value when training starts, and should be gradually decreased over time. This would enable a large exploration of the parameter space in the beginning of training, and focus on surrounding the optimum towards the end of training.

Adaptive learning-rate methods (a.k.a optimizers) have been proposed to alleviate these issues. Adagrad [41], RMSprop [161], and Adaptive Moment Estimation (Adam) [94] are such optimizers. Adam is often cited as one of the most efficient optimizers and is extensively used in our experiments.

2.3.5 Evaluation protocol

KGEM performance is usually evaluated in two stages. During the validation phase, KGEM performance is evaluated on the validation set $\mathcal{T}_{valid}$ after regular – often uniform – intervals of epochs. This way, the best epoch is identified. During the test phase, KGEM performance is ultimately evaluated on the test set $\mathcal{T}_{test}$ after retrieving the optimal model parameters achieved at the best epoch of validation. Whether during the validation or test phase, KGEM performance is evaluated the same way: sifting through every triple of the test (resp. validation) set, both head and tail predictions are performed. In the case of head prediction, this amounts to taking a triple (h, r, t) from the test (resp. validation) set, hiding the ground-truth head entity – resulting in $(?, r, t)$ – and letting the KGEM assign a score to every possible entity as a candidate for the head position. These scores are finally ordered and reflect the plausibility of such facts. Tail prediction is performed analogously on $(h, r, ?)$.

In both cases, the rank of the ground-truth entity from the test (resp. validation) set is used to compute aggregated rank-based metrics based on the top-K scored entities. The rank of the ground-truth entity can be determined in two different ways that depend on how observed facts – *i.e.* facts that already exist in the KG – are considered. In the raw setting, observed facts outranking the ground-truth are not filtered out, while this is the case in the filtered setting. For instance, assuming head prediction is performed on the given ground-truth triple (`BarackObama`, `livesIn`, `USA`), a KGEM may assign a lower score to this triple than to the following triple: (`MichelleObama`, `livesIn`, `USA`). The latter triple actually represents an observed fact. In the raw setting, this triple would not be filtered out from top-K scored triples. This can cause the evaluation procedure to not properly assess the KGEM performance. This is why in practice, the filtered setting is commonly preferred. In the present work, the filtered setting is also used.

KGEM performance is almost exclusively assessed using the following rank-based metrics: Hits@K, Mean Rank (MR), and Mean Reciprocal Rank (MRR) [70]. In the following, we recall their definitions.

Hits@K (Equation 2.11) accounts for the proportion of ground-truth triples appearing in the first K top-scored triples:

$$\text{Hits@}K = \frac{1}{|\mathcal{T}|} \sum_{q \in \mathcal{T}} \mathbb{1}\left[\text{rank}(q) \leq K\right] \tag{2.11}$$

where $\mathcal{T}$ is the batch of ground-truth triples, $\text{rank}(q)$ is the position of the ground-truth triple q in the sorted list of triples, and $\mathbb{1}\left[\text{rank}(q) \leq K\right]$ yields 1 if q is ranked between 1 and K, 0 otherwise. This metric is bounded in the $[0, 1]$ range and its values increase with K, where the higher the better. In the literature, the most frequent choices for the K values are $K = 1$ and $K = 10$, plus a third intermediate value usually chosen to be $K = 3$ or $K = 5$. In this thesis, we stick to this common choice by systematically reporting results in terms of Hits@1 and Hits@10, with some flexibility regarding the intermediate value (either Hits@3 or Hits@5).

Mean Rank (MR) (Equation 2.12) corresponds to the arithmetic mean over ranks of ground-truth triples:

$$\text{MR} = \frac{1}{|\mathcal{T}|} \sum_{q \in \mathcal{T}} \text{rank}(q) \tag{2.12}$$

This metric is bounded in the $[0, |\mathcal{E}|]$ interval, where $|\mathcal{E}|$ stands for the number of entities in the KG, where the lower the better.

Mean Reciprocal Rank (MRR) (Equation 2.13) corresponds to the arithmetic mean over reciprocals of ranks of ground-truth triples:

$$\text{MRR} = \frac{1}{|\mathcal{T}|} \sum_{q \in \mathcal{T}} \frac{1}{\text{rank}(q)} \tag{2.13}$$

Contrary to MR, MRR is a metric bounded in the $[0, 1]$ interval. Higher results indicate better performance. Because this metric does not use any threshold K compared to Hits@K, it is less sensitive to outliers. In addition, it is often used for performing early stopping and for tracking the best epoch during training [10, 70].

Beyond the choice of metrics, the evaluation procedure itself matters and can lead to too optimistic or pessimistic results [156]. In particular, the strategy for breaking ties when triples are assigned the same score should be defined beforehand. Three main strategies exist:

- **Bottom (pessimistic)**: assumes that the ground-truth is on the last position of all triples with equal score
- **Top (optimistic)** assumes that the ground-truth is on the first position of all triples with equal score.
- **Random (realistic)** assumes that the ground-truth is randomly placed among the triples with equal score. Alternatively, under expectation it can be seen as the mean of the optimistic and the pessimistic rank[9].

Both top (optimistic) and bottom (pessimistic) strategies are biased and the random (realistic) strategy should be the go-to approach [156]. Consequently, we use the latter strategy in all our experiments.

2.4 A deep dive into popular embedding-based models

In this section, we provide an in-depth description of popular KGEMs. As mentioned before, the number of KGEMs proposed in the literature is substantial. Consequently, we only provide details about the KGEMs we will extensively used in the following chapters of this thesis.

TransE [18] is the earliest translational model. It learns representations of entities and relations such that for a triple (h, r, t), $\mathbf{e}_h + \mathbf{e}_r \approx \mathbf{e}_t$, where $\mathbf{e}_h$, $\mathbf{e}_r$ and $\mathbf{e}_t$ are the head, relation and tail embeddings, respectively. The scoring function is

$$f(h, r, t) = d(\mathbf{e}_h + \mathbf{e}_r - \mathbf{e}_t) \tag{2.14}$$

where d is a distance function, usually the $L1$ or $L2$ norm.

TransH [174] is an extension of TransE. It allows entities to have distinct representations when involved in different relations. Specifically, $\mathbf{e}_h$ and $\mathbf{e}_t$ are projected into relation-specific hyperplanes with projection matrices $\mathbf{w}_r$. If (h, r, t) holds, the projected entities $\mathbf{e}_{h_\perp} = \mathbf{e}_h - \mathbf{w}_r^T \mathbf{e}_h \mathbf{w}_r$ and $\mathbf{e}_{t_\perp} = \mathbf{e}_t - \mathbf{w}_r^T \mathbf{e}_t \mathbf{w}_r$ are expected to be linked by the relation-specific translation vector $\mathbf{d}_r$. Thus, the scoring function is

$$f(h, r, t) = d(\mathbf{e}_{h_\perp} + \mathbf{d}_r - \mathbf{e}_{t_\perp}) \tag{2.15}$$

[9] https://pykeen.readthedocs.io/en/stable/tutorial/understanding_evaluation.html

TransH often showcases better performance than TransE with only slightly more parameters [174].

DistMult [181] is a semantic matching model. It is characterized as such because it uses a similarity-based scoring function and matches the latent semantics of entities and relations by leveraging their vector space representations. More specifically, DistMult is a bilinear diagonal model that uses a trilinear dot product as its scoring function:

$$f(h, r, t) = \langle \mathbf{e}_h, \mathbf{W}_r, \mathbf{e}_t \rangle \tag{2.16}$$

It is similar to RESCAL [122] – the very first semantic matching model – but restricts relation matrices $\mathbf{W}_r \in \mathbb{R}^{d \times d}$ to be diagonal.

ComplEx [164] is also a semantic matching model. It extends DistMult by using complex-valued vectors to represent entities and relations: $\mathbf{e}_h, \mathbf{e}_r, \mathbf{e}_t \in \mathbb{C}^d$. As a result, ComplEx is better able to model antisymmetric relations than DistMult [157]. Its scoring function uses the Hadamard product:

$$f(h, r, t) = \mathrm{Re}\left(\mathbf{e}_h \odot \mathbf{e}_r \odot \overline{\mathbf{e}}_t\right) \tag{2.17}$$

where $\overline{\mathbf{e}}_t$ denotes the conjugate of $\mathbf{e}_t$.

SimplE [93] models each fact in both a direct and an inverse form. To do so, an entity e is simultaneously represented by two vectors $\mathbf{e}^h, \mathbf{e}^t \in \mathbb{R}^d$. Depending on whether e appears as head or tail in a given triple, either $\mathbf{e}^h$ or $\mathbf{e}^t$ is used. Consequently, the two entity representations $\mathbf{e}^h$ and $\mathbf{e}^t$ are learned independently. Likewise, each relation r comes with a direct and an inverse vectors $\mathbf{e}_r$ and $\mathbf{e}_{r^{-1}}$. The scoring function reflects the interaction between all the aforementioned entity and relation embeddings:

$$f(h, r, t) = \frac{1}{2}(\left\langle \mathbf{e}_h^h, \mathbf{e}_r, \mathbf{e}_t^t \right\rangle + \left\langle \mathbf{e}_h^t, \mathbf{e}_{r^{-1}}, \mathbf{e}_t^h \right\rangle) \tag{2.18}$$

ConvE [39] first reshapes entity and relation embeddings and then concatenates them into a 2D matrix [h; r]. To model the interactions between entities and relations, ConvE subsequently uses 2D convolution over embeddings and layers of nonlinear features. The output is ultimately scored against the tail embedding t using the dot product. More precisely, the following scoring function is used:

$$f(h, r, t) = g\left(\mathrm{vec}\left(g\left(\mathrm{concat}\left(\widehat{\mathbf{e}}_h, \widehat{\mathbf{e}}_r\right) * \omega\right)\right) \mathbf{W}\right) \cdot \mathbf{e}_t \tag{2.19}$$

where g denotes a non-linear function, vec is the vectorization operation reshaping a tensor into a vector, concat is the concatenation operator, $*$ and . denote a convolution and a dot product, respectively, $\widehat{\mathbf{e}}$ denotes a 2D reshaping of e and ω is the set of convolutional filters.

ConvKB [120] also represents entities and relations as same-sized vectors. However, ConvKB does not reshape the embeddings of entities and relations. Plus, ConvKB also considers the tail embedding in the concatenation operation, thus obtaining the 2D matrix [h; r; t] after concatenation. Convolution by a set ω of T filters of shape $1 * 3$ is applied on this input. The resulting $T*3$ feature map then passes through a dense layer with one neuron and a weight matrix W. Finally, the following scoring function assesses the plausibility of a given triple (h, r, t):

$$f(h, r, t) = \mathrm{concat}\left(g\left([\mathbf{e}_h, \mathbf{e}_r, \mathbf{e}_t] * \omega\right)\right) \cdot \mathbf{w} \tag{2.20}$$

It has been claimed that the concatenation of a set of feature maps generated by convolution should increase the learning ability of latent features compared to ConvE [90]. However, recent works point out the evaluation procedure used in the original implementation of ConvKB [149, 156], which may result in overly optimistic results on the benchmark datasets FB15K237 and WN18RR. Nonetheless, due to the popularity of this model, we choose to include it in our experiments.

R-GCN [147] extends the idea of applying graph convolutional networks (GCNs) to multi-relational data. R-GCN operates on local graph neighborhoods and applies a convolution operation to the neighbors of each entity. By aggregating the messages coming from all the neighbors of an entity, the embedding of the latter is updated in accordance. Each entity thus has a hidden representation which directly depends on the hidden representations of its neighbors. This process of accumulating messages (*i.e.* the hidden representations of neighboring entities) and aggregating them so as to update the hidden representation of the central node is performed for each layer of the R-GCN model. More formally, the hidden representation of the entity i in the layer $(l+1)$ is defined as:

$$h_i^{(l+1)} = \sigma\left(W_0^{(l)} h_i^{(l)} + \sum_{r \in R} \sum_{j \in N_i^r} \frac{1}{c_{i,r}} W_r^{(l)} h_j^{(l)}\right) \tag{2.21}$$

where $\sigma(.)$ can be any element-wise activation function, N_i^r denotes the set of neighboring entities of entity i considering the relation r, $W_r^{(l)}$ is the relation-specific weight matrix for layer l and $c_{i,r}$ is a normalization constant. To update entity hidden representation taking into account its previous state, a self-connection is also incorporated into the activation function $\sigma(.)$. Such layers can be stacked multiple times in order to better learn interactions and dependencies across several relational steps. However, as applying Equation(2.21) directly would dramatically increase the number of parameters for KGs having lots of different relation types, basis-decomposition and block-diagonal-decomposition are proposed in [147] to reduce model parameter size and prevent overfitting. In the case of basis decomposition which is used in the original paper, the relation-specific weight matrix $W_r^{(l)}$ is decomposed into a linear combination of basis transformation $V_b^{(l)}$ with coefficients $a_{rb}^{(l)}$:

$$W_r^{(l)} = \sum_{b=1}^{B} a_{rb}^{(l)} V_b^{(l)} \tag{2.22}$$

CompGCN [166] improves over R-GCN by not only learning entity representations but also relation representations. Concretely, CompGCN performs a composition operation $\phi(.)$ over each edge in the neighborhood of a central node. The composed embeddings are subsequently convolved with direction-specific weight matrices W_O and W_I – for original and inverse relations, respectively. In addition to the entity representation update, relation representations are also updated individually:

$$h_r^{(l+1)} = W_{rel}^{(l)} h_r^{(l)} \tag{2.23}$$

where W_{rel} is a learnable transformation matrix which projects all the relations into the same embedding space as entities.

2.5 Mainstream datasets for link prediction

Datasets for benchmarking LP in the transductive setting are usually obtained by sampling real-world KGs (*e.g.*, the ones presented in Section 2.1) and splitting the obtained triples into a training, a validation and a test set and ensuring that entities and relations in the validation and test sets also appear in the training set (see Section 2.3. Numerous datasets are utilized in experimental procedures. They either derive from general KGs (*e.g.* DBpedia, Freebase, Wikidata, YAGO), domain-specific KGs (*e.g.* BioKG, EduKG, Hetionet, PharmKG), or are not underpinned by any particular schema (*e.g.* Countries, Kinships, Nations). Although this is not the focus of the present work, we nevertheless mention the existence of a broader spectrum of LP datasets, covering specific applications such as inductive LP [73], temporal LP [26], LP with literals [67] and LP with n-ary facts [47].

Despite the handful of proposed datasets, only a few of them are well-established benchmark. In the following, we consequently focus on their respective descriptions:

FB15k [18] was built by selecting all the Freebase entities with more than 100 mentions and extracting the triples involving them (thereby including their neighbors), excepts triples containing literals such as dates and numerical quantities. N-ary relations were also reified. FB15k was by far the most used dataset for many years, but it was soon demonstrated to suffer from data leakage [163], as numerous triples in the test set could be recovered by taking the inverse relation of training triples and swapping the subject and object.

FB15k-237 [163] is a subset of FB15k that was obtained by first limiting the set of relations to the most frequent 401 relations and then filtering out all the equivalent or inverse relations. The number of relations was consequently decreased from 1,345 to 237. In addition, Toutanova *et al.* removed all triples in validation and test sets whose subjects and objects were connected in the training set through any relation. It was later shown that this step could unduly remove useful information [2].

WN18 [18] originates from WordNet. WordNet entities are *synsets* (word senses) and relations denote their lexical dependencies (e.g. "hypernym"). To build WN18, Bordes *et al.* leveraged WordNet as a whole and subsequently filtered out entities and relationships with too sparse mentions.

WN18RR [39] originates from WN18. As for FB15k, Toutanova *et al.* [163] reported a huge test leakage in the original WN18 dataset. More specifically, 94% of the train triples have inverse relations that are linked to test triples. Dettmers *et al.* [39] remove all inverse relations to propose WN18RR. In WN18RR, entities are nouns, verbs, and adjectives.

YAGO3-10 [108] is a subset of YAGO3 – being itself an extension of YAGO – which was built by sampling entities associated with at least 10 different relations and all the triples where they appear (thereby also including neighbors). The majority of its triples describe attributes of persons such as citizenship, gender, and profession. The relatively lower performance of the Inverse Model [39] over YAGO3-10 imply that this dataset is more robust to test leakage than FB15k and WN18.

These five datasets are the most frequently used in recent publications. In Chapter 3.2.2, we will see how this thesis goes a step beyond their mere use, by enriching them with schema-based

information with the goal of developing neuro-symbolic training approaches – especially for link prediction.

Chapter 3

Overcoming data limitations in knowledge graph-based applications with semantically rich datasets

Contents

This chapter covers the following articles:

PyGraft: Configurable Generation of Synthetic Schemas and Knowledge Graphs at Your Fingertips. Hubert *et al.* (2024). The Semantic Web - 21st International Conference, ESWC 2024, Hersonissos, Crete, Greece, May 26 - May 30, 2024. (**Best Resource Paper Award**)

New Ontology and Knowledge Graph for University Curriculum Recommendation. Hubert *et al.* (2022). Proceedings of the ISWC 2022 Posters, Demos and Industry Tracks: From Novel Ideas to Industrial Practice, Virtual Conference, Hangzhou, China, October 23-27, 2022.

Vers un système de recommandation explicable pour l'orientation scolaire. Hubert *et al.* (2022). Workshop EXPLAIN'AI-EGC 2022, Blois, France, January 24-28, 2022.

3.1 Motivations

Chapter 2 explored the task of LP over KGs, utilizing machine learning models that learn patterns from data. Despite their effectiveness, these models often fail to exploit the full spectrum of information available within KGs. This observation leads us to the heart of neuro-symbolic AI approaches, which aim to combine statistical methods with symbolic knowledge. Recent works highlighted the effectiveness of incorporating schema-based information during model training [87, 111, 35]. However, a notable gap exists in the availability of schema-enriched datasets for link prediction. Prominent datasets such as FB15k237 and YAGO3-10, although comprehensive, typically exclude schema-based information. This requires a cumbersome process where end users must manually extract and integrate this information. Addressing this concern, a primary contribution of this thesis is the semantic enrichment of benchmark datasets for link prediction, enhancing the accessibility and advancement of neuro-symbolic representation learning.

The utility of KGs extends beyond the LP task and finds applications in specific applications such as recommender systems. However, domain-specific applications often grapple with the scarcity of relevant, publicly available data, particularly in the sensitive field of education and school guidance. In the framework of the project AILES, our work consequently pivots to the educational domain, where we aim to develop an explainable recommender system to guide high school students in their university choices. Confronted with the lack of suitable resources, we initiated a comprehensive data collection campaign, resulting in the creation of EducOnto and EduKG – an ontology and a knowledge graph for curriculum recommendation. This endeavor, though fruitful, highlighted the labor-intensive and domain-specific nature of such data collection.

To broaden the scope and applicability of our research, we developed PyGraft, a Python-based tool for generating synthetic ontologies and KGs. Notably, PyGraft addresses the prevalent issues of data scarcity and evaluation bias in the research community. While access to domain-specific public data is often restricted, PyGraft facilitates the creation of datasets that, although synthetic, are valuable for prototyping and testing new approaches. The significance of this

tool lies in its ability to mimic the topological and semantic features of real-world data, thereby offering a more diverse and unbiased platform for evaluating new models and techniques. This approach is particularly relevant given recent findings that a multitude of benchmarks does not necessarily translate to diverse characteristics, leading to potential biases in model evaluation.

This chapter discusses the three aforedescribed issues and how this thesis contributes to mitigate them, thereby addressing **RQ1** (Section 1.1), which was defined as follows:

> **RQ1.** How can we propose publicly available and semantically rich resources to foster the development of neuro-symbolic approaches for KG-based applications, *e.g.* link prediction?

In particular, to answer the more complex **RQ1**, we subdivide it into sub research questions, on the basis of the previously described current state of research:

> **RQ1.1.** How can we enrich existing datasets for link prediction?
>
> **RQ1.2.** In the specific domain of curriculum recommendation and educational guidance, can we propose semantically rich resources to model the transition from high school to university, in an attempt to develop an explainable recommender systems?
>
> **RQ1.3.** How can we go beyond the manual curation of semantically rich resources, and propose automatically generated and synthetic resources to help researchers prototype and evaluate their approaches more comprehensively?

These contributions collectively respond to a fundamental concern in the field: the need for more accessible, diverse, and representative datasets to foster the development of robust and effective neuro-symbolic approaches for KG-based applications. Each of the above sub research questions is addressed in a dedicated section, which uniquely contributes to answering **RQ1**. In particular, this chapter presents the following contributions:

1. Mainstream datasets for the link prediction task are semantically enriched with schema-based information. These augmented datasets are made publicly available to foster the development of neuro-symbolic representation learning, especially in the context of link prediction (Section 3.2) (**RQ1.1**).
2. We propose EducOnto and EduKG – an ontology and a knowledge graph in the educational domain. Although these resources are primarily concerned with the task of curriculum recommendation, they can be used in a broad range of education-related tasks (Section 3.3) (**RQ1.2**).
3. We develop and release PyGraft – a Python-based tool for generating synthetic ontologies and KGs. In particular, PyGraft broadens our previous attempt at providing domain-specific resources but abstracts the semantic layer to instead focus on the issues of diversity and reproducibility (Section 3.4) (**RQ1.3**).

3.2 Semantic data enrichment for neuro-symbolic approaches

As previsouly mentioned, this thesis is concerned with developing neuro-symbolic approaches to perform LP with additional, schema-based knowledge. Consequently, such knowledge should be readily available and used by the KGEMs in par with factual triples. However, the LP benchmarks described in Section 2.5 either rely on an instance view or an ontology view, but not the two simultaneously [64]. Therefore, for datasets only composed of relational triples, some engineering is needed to retrieve ontological information about entities and predicates from the schemas that underpin these datasets.

3.2.1 Bridging ontologies and knowledge graphs: a few attempts

A few works are concerned with the same need and thus extracted relevant schema-based information to be merged with relational knowledge.

In [179], an enriched version of FB15k is proposed. Entity types are extracted by querying the `type/instance` field in Freebase, and relation profiles (*i.e.* domains and ranges) are similarly extracted by querying the `rdf-schema#domain` and `rdf-schema#range` fields, respectively. Entity types that never occur as either domain or range of any relation are filtered out. 571 entity types are ultimately kept, with all entities in the dataset having at least one type and the average number of types per entity is roughly 8.

Moon *et al.* [116] build a similar typed-version of FB15k with entity types and relation-specific information, with subtle differences compared to [179]. Their dataset is names FB15kET to denote the presence of entity types. By the same token, they extract relational triples from YAGO3 to build the schemaless YAGO43k dataset, that they further enrich with entity types and relation domains and ranges to propose YAGO43kET.

[107] also builds schema-enriched datasets out of YAGO3, that they call YAGO39k and M-YAGO39k. These datasets contain `instanceOf` and `subClassOf` relations, but do not encompass any relation-specific information. M-YAGO39k is itself an extension of YAGO39k with additional triples in the validation and test sets that are generated based on YAGO39k using the transitivity of `isA` relations (*e.g.* if `AssociateProfessor` is a `subClassOf AcademicStaff`, and that `Bob` is known to be an `AssociateProfessor`, then the information that `Bob` is also an `AcademicStaff` is added to M-YAGO39k).

YAGO26K-906 and DB111K-174 are two recent schema-enriched LP datasets presented in [64]. The authors first sample a random subset of relational triples from YAGO3 and DBpedia, respectively. After obtaining the entity set of their "instance graph", they extract cross-alignment of the retrieved entities in YAGO3 and DBpedia. Entities are therefore associated to concepts or classes, that will correspond to the nodes in their "ontology graph". This ontology graph is built by taking the concepts from the last step, and extracting the subgraphs around them in their original ontology. However, we note that the resulting ontology graphs are rather scarce: while the instance graph of DB111K-174 contains 111,762 entities, 305 relations, and 863,643 triples, its respective ontology graph only contains 174 concepts and 763 triples. In addition, the presence of only 99,748 `instanceOf` relations suggests that a relatively high number of entities are actually untyped.

3.2.2 Proposed datasets for schema-based representation learning

In Sections4 and 5, we discuss our results performed on a broad set of schema-enriched datasets. Specifically, the majority of them originate from our own research and are therefore an integral part of this thesis work. Further details about these datasets are provided below.

AIFB [13] is a Semantic Web (RDF) dataset originally used as a benchmark in data mining. It records the organizational structure of AIFB at the University of Karlsruhe. However, this dataset also lends itself to prototyping LP models due to the presence of relational triples and its relatively small-size. In our experiments, we use AIFB in the LP framework after proper removal of triples featuring literals (*e.g.* ages and dates) and after processing the dataset in such a way that relational triples are distinguished from the schema.

CoDEx-S/M [146] are datasets extracted from Wikidata and Wikipedia. They cover a wider

scope and purposely contain harder facts than most KGs [146]. Consequently, these datasets prove to be more challenging for the LP task. Codex-S and Codex-M contain entity types, relation descriptions and Wikipedia page extracts. Nonetheless, Wikidata lacks `rdfs:domain` or `rdfs:range` predicates, making it more challenging to handle property constraints.

DB93k [79] is a subset of DB100K, which was first introduced in [40]. A slightly modified version of DB100K has been proposed in [35]. Contrary to the initial version of DB100K, the latter version is schema-defined: entities are properly typed and most relations have a domain and/or a range. This second version is considered in the following experiments. However, some inconsistencies were found in the dataset. Some DBpedia entities only instantiate Wikidata[10] or `schema.org`[11] classes, while instantiation of classes from the DBpedia ontology actually exist. Moreover, some entities are only partially typed. It must also be noted that domains and ranges of relations have been extracted from DBpedia more than two years ago. DBpedia is a communautary and open-source project: DBpedia classification system relies on human curation, which sometimes implies a lack of coverage for some resources and updates for others. Consequently we associated all relations of DB100K to their most up-to-date domains and ranges[12]. Similarly for entities, we associated all entities to their most up-to-date classes. Finally, we removed all untyped entities and we enforced that validation and test sets contain triples whose relation have a well-defined domain (resp. range), as well as more than 10 possible candidates as head (resp. tail). This ensures that the newly semantic-oriented Sem@K metric presented in Section 4 is not unduly penalized and can be calculated on the same set of entities as Hits@K and MRR, at least until $K = 10$.

DB77k [80] is a refined version of the previously described DB93k dataset. Compared to DB93k, we add the requirement that both the head h and tail t of train triples have another semantically valid counterpart for negative sampling. This prerequisite – in addition to the filtering operations on DB93k – guarantee that each positive train triple can be paired with at least one semantically valid triple. This will be of particular importance when discussing loss functions semantically enriched with domain and range constraints for LP (Section 5).

DBpedia50 [150] does not originally come with entity-type information[13]. However, we ran SPARQL queries against DBpedia to get entity types for all entities in the dataset.

FB15k237-ET [79] derives from FB15k237 [163]. To the best of our knowledge, there is no schema-defined version of FB15k237. However, as previously mentioned there exists schema-defined versions of FB15k [179, 116]. We based ourselves on the schema-enriched version of FB15k237 found in [179] and mapped the extracted entity types, relation domains and ranges to the entities and relations found in FB15k237. The resulting schema-defined version of FB15k237 is named FB15k237-ET and includes only typed entities. Besides, validation and test sets contain triples whose relation have a well-defined domain (resp. range), as well as more than 10 possible candidates as head (resp. tail).

FB15k187 [80] is a refined version of the previously described FB15k237-ET dataset obtained

[10] `https://www.wikidata.org/`
[11] `https://schema.org/`
[12] SPARQL queries were fired against DBpedia as of November 9, 2022
[13] `https://github.com/bxshi/ConMask`

Table 3.1: Datasets used in this thesis work. Column headers from left to right: name of the dataset, papers in which the dataset is used, number of entities, relations, classes, training triples, validation triples, and test triples

Dataset	Papers	$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{R}\rvert$	$\lvert\mathcal{C}\rvert$	$\lvert\mathcal{T}_{train}\rvert$	$\lvert\mathcal{T}_{valid}\rvert$	$\lvert\mathcal{T}_{test}\rvert$
AIFB	[74]	2,389	16	18	14,170	745	785
CoDEx-S	[79, 74]	2,034	42	502	32,888	1,827	1,827
CoDEx-M	[79, 74]	17,050	51	1,503	185,584	10,310	10,311
DB93k	[79]	92,574	277	311	237,062	18,059	36,424
DB77k	[80, 78]	76,651	150	280	140,760	16,334	32,934
DBpedia50	[74]	24,624	351	285	32,388	123	2,098
EduKG	[76, 75]	5,452	27	9	25,411	4,279	4,258
FB15k237-ET	[75, 79, 74]	14,541	237	643	272,115	17,535	20,466
FB15k187	[80, 78]	14,305	187	624	245,350	15,256	17,830
KG20C	[76, 75]	16,362	5	5	48,213	3,670	3,724
YAGO3-37k	[79]	37,335	33	132	351,599	4,220	4,016
YAGO4-19k	[79, 74]	18,960	74	1,232	27,447	485	463
YAGO4-14k	[80, 78]	14,178	37	954	18,263	472	448

by applying the same filtering step as for DB77k compared to DB93k.

YAGO3-37k [79] derives from the YAGO39k dataset [107] previously described. Compared to the original YAGO39k, in our experiments only typed entities are kept. In addition, relations having less than 10 observed heads or tails in the training set are discarded from the validation and test splits, for the same reason that keeping them would not reflect the actual Sem@K values. It should be noted that in the YAGO3 ontology, most relations have very generic domains and ranges which are very close to the root of the ontology hierarchy. To produce a more challenging evaluation setting of the models' semantic awareness (Section 4), a subset of hard relations was identified and only validation and test triples whose relation belongs to this subset are kept. The resulting dataset is named YAGO3-37k.

YAGO4-19k [79] was manually built from scratch based on the new YAGO4 KG [158]. Similarly to other datasets, we focused on relations with a defined domain and range, and more than 10 triples to constitute the validation and test sets. We purposely favored difficult relations to feature in the validation and test sets. To enrich the training set, additional relations were added based on a manual selection. Selected relations in validation and test sets as well as additional relations in the training set can be found on GitHub[14]. It should be noted that the class hierarchy associated with entities in YAGO4 is `schema.org`.

YAGO4-14k [80] is a refined version of the previously described YAGO4-19k dataset obtained by applying the same filtering step as for DB77k compared to DB93k.

[14] `https://github.com/pmonnin/YAGO4-LP`

3.3 An ontology and a knowledge graph in the educational domain

Our proposed schema-based datasets (see Section 3.2.2) – although augmented with semantically rich information about entities and relations – are still part of existing and general-domain KGs.

However, in some real-world applications, it makes more sense to use domain-specific data. Recall that this work is concerned with the specific task of recommending curricula to high school and university students. Consequently, in what follows we provide the reader with background knowledge and the motivations that underpin the project AILES.

Recommender systems (RSs) have traditionally been designed using collaborative filtering or content-based approaches [14]. The more specific domain of educational RSs has long followed this same pattern [117, 25]. However, recommending curricula with the sole consideration of students' profiles and/or peers' choices may lead to inaccurate or insufficiently tailored recommendations [125]. This is mainly because educational guidance is one of such high stake domains in which students must make their decisions carefully, especially when it comes to picking a university curriculum. In this thesis, we aim at going beyond the traditional collaborative filtering and content-based approaches and fully leverage the power of ontologies and KGs to enhance the quality of recommendations, as it has successfully been done in recent works more broadly concerned with university course recommendations [81, 82, 125, 180]. However successful these attempts are, they are only concerned with recommending university modules to students already enrolled at university. Our aim is somewhat different: we want to bridge the gap between high school and university, by providing school guidance about which choices high school students should consider for their future academic path in the university track. Unfortunately, no available ontologies or datasets relate to this specific application, which is nonetheless a turning point in student's lives. Facing the need for publicly available resources about educational guidance and curricula choices to further develop an explainable recommender system, we built our own educational ontology and KG – named EducOnto and EduKG, respectively [71].

3.3.1 EducOnto

The purpose of this section is to elaborate on the methodology we used to design EducOnto. We followed best practices guidelines presented in [124]. Consequently, three main phases are distinguished: (1) Domain and purpose definitions; (2) Ontology building process; (3) Ontology evaluation. The exact steps taken to formalize EducOnto are described below. EducOnto is intended to be highly reusable and for this reason we make it publicly available at the following link: purl.org/educonto.

Domain and Purpose Definitions

In the educational domain and more particularly the transition from high school to university, we aimed at identifying the core concepts and relationships that are key to understanding how curricula are organized, what drives students to choose a given academic path, and what are the intermediate influential factors. In Section 3.3.1 we elaborate on the main classes and properties we identified. EducOnto formalizes knowledge about them so as to ensure the fulfilment of several educational-related, downstream tasks. One use case we are particularly interested in is recommending university curricula in which students are expected to fulfil themselves. To make sure that the scope of EducOnto is well-defined and that resources built on top of EducOnto are able to answer specific needs and requirements, we enumerate a list of competency questions.

These competency questions are presented in Section 3.3.1.

Ontology Building Process

As previously mentioned, we follow the widely used method proposed in [124] to build EducOnto.

(i) Reusing existing ontologies. A first step consists in reusing as much as possible existing ontologies. We notice that very few ontologies about higher education are publicly available. A thorough university ontology is available on the University of Maryland's Computer Science Department's homepage[15]. However, this ontology is rather old and does not take into the individual characteristics we are interested in, such as the favorite school subjects of students or their center of interests. This ontology models the university and research environment and lacks information about the offered curricula. The ontology proposed in [38] could have been of interest. However it is not maintained anymore and does not model students' profiles. The same remark applies to the ontology introduced in [92]. Even though it provides us with interesting terms and concepts that can be considered for building EducOnto, students' profiles are insufficiently filled. Conversely, the ontology found in [83] puts the emphasis on students' profiles. But their domain and purpose definitions are different from ours. By contrast, Obeid *et al.* present a preliminary version of an ontology in [125]. Their domain and purpose definitions are very similar to ours. In their approach, they also consider personality and individual factors such as the favorite school subjects. As a result, we chose to base ourselves on this preliminary work.

(ii) Listing important terms. We subsequently enumerate all the important terms related to our application domain. This was done in collaboration with psychologists from the University of Lorraine, France, that are part of the project AILES. Two main concepts stand out: student and curriculum. Both concepts are related to other intermediate concepts: major, specialty, high school diploma, school subject, center of interest and so on. We also list geographical and demographic concepts such as gender, age, high school location and university location. In order to respect the confidentiality of students included in EduKG, we choose not to integrate any geographical and demographic concepts into EducOnto.

(iii) Defining classes and properties. Adopting a top-down development process, we end up with 71 relevant classes, including 62 subclasses. Examples of such classes are: `Curriculum`, `FieldOfStudy` and `HighSchoolMajor`. Let us now focus on the definition of relevant properties for the aforementioned classes. Properties do not hold independently of the classes they are associated with. Properties are divided into two categories: data properties and object properties. Data properties link data to individuals (i.e. class instances) whereas object properties bond individuals together via a property [49]. Regarding our application domain, data properties would be the natural choice for modelling such information as age, gender and so on. But as this information is to stay anonymous, we decided to focus on object properties. In the end, 30 object properties were identified. A general overview of EducOnto that contains the main classes and properties is presented in Figure 3.1.

Table 3.2 lists some of the main classes of EducOnto and two object properties are briefly presented in Table 3.3. In both tables, the `educonto` prefix stands for http://purl.org/educonto/ and it is used to avoid specifying the full names of EducOnto classes and properties. For a more comprehensive presentation of EducOnto, documentation is made available at the following link: https://nicolas-hbt.github.io/educ-ontokg/widoco/index-en.html.

[15] `http://www.cs.umd.edu/projects/plus/SHOE/onts/univ1.0.html#ext`

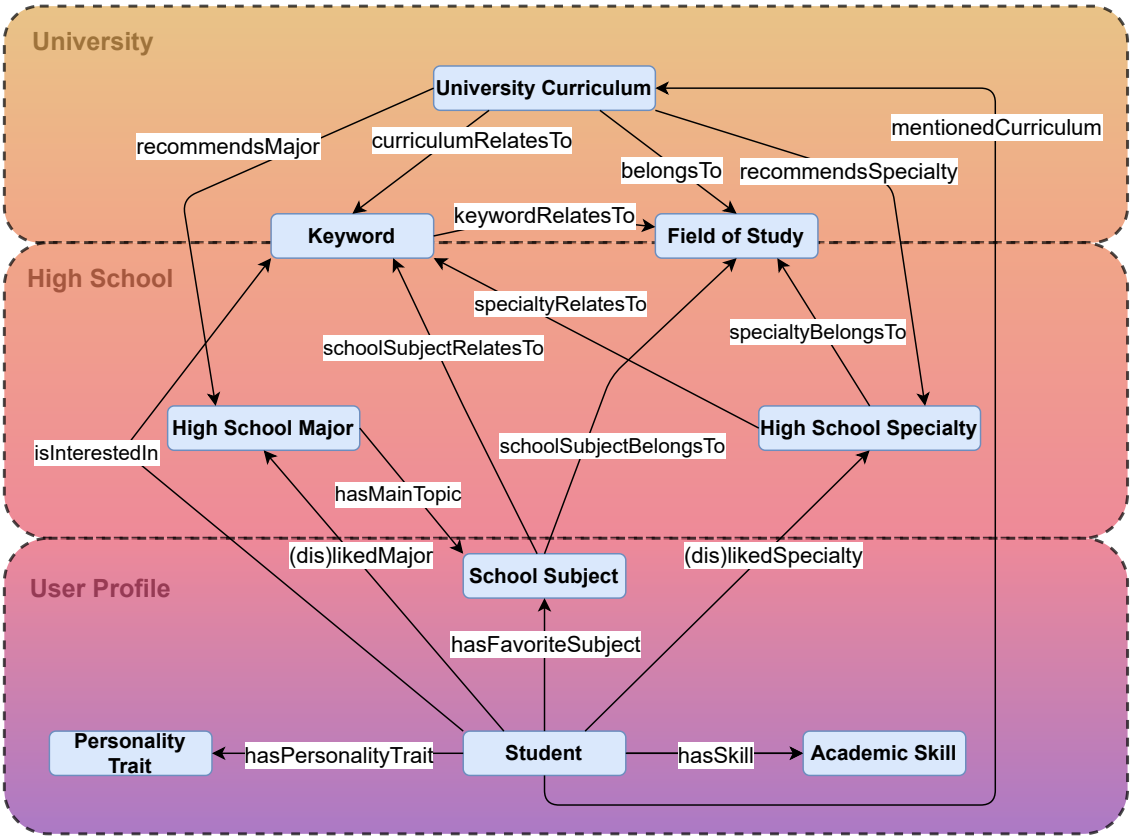


Figure 3.1: A conceptual overview of EducOnto. The lower, middle and upper parts of the diagram depict the classes and properties related to the User Profile, High School and University, respectively

Table 3.2: EducOnto's main classes with a description and an example of instance

Class	Description	Instance Example
`educonto:AcademicSkill`	Skill held by a student	<http://purl.org/edukg/good_memory>
`educonto:Curriculum`	University curriculum	<http://purl.org/edukg/curriculum/lg_psycho>
`educonto:FieldOfStudy`	Field of study	<http://purl.org/edukg/engineering>
`educonto:Keyword`	Mentioned keyword	<http://purl.org/edukg/healthcare>
`educonto:Person`	Student	<http://purl.org/edukg/stud/user999>
`educonto:PersonalityTrait`	Personality trait	<http://purl.org/edukg/investigative>
`educonto:SchoolSubject`	School subject	<http://purl.org/edukg/chemistry>

Table 3.3: Two EducOnto object properties

Object Property	Domain	Range
`educonto:hasSkill`	`educonto:Person`	`educonto:AcademicSkill`
`educonto:hasPersonalityTrait`	`educonto:Person`	`educonto:Keyword`

Ontology Evaluation

By definition, an ontology is an arbitrary construction of a domain. However subjective this construction might be, this does not diminish the need for an objective and rational evaluation. Five ontology evaluation methods can be distinguished in the literature [68, 85, 139]:

- Gold standard evaluation compares the proposed ontology with an already existing one which acts as a reference model;
- Criteria-based evaluation specifically assesses the quality of the proposed ontology based on predefined criteria of interest;
- Data-driven evaluation implies the direct comparison of the proposed ontology with real data from the domain of interest;
- Application-based (also called task-based) evaluation is concerned with the efficiency of the proposed ontology in the context of real-world use cases for which the ontology was firstly designed;
- Human evaluation involves field experts, end users or lay humans to assess the quality of the proposed ontology.

We argue that education is a high stakes domain which requires expert input. Moreover, in terms of educational guidance there are many requirements and constraints that are not laid down formally. Education experts are aware of such unwritten rules. This is why we firstly choose to develop and evaluate EducOnto in collaboration with experts in educational sciences. This allows us to disambiguate some terms and to both clarify and enrich the structure of the ontology. For instance, the initial version of EducOnto did not reflect the whole spectrum of university curricula. Vocational Bachelor's Degrees were not considered, and the main `Curriculum` class was oversimplified in its internal hierarchy of subclasses. Expert evaluation allowed us to continuously improve the modelling scope of EducOnto. As previously mentioned, EducOnto was built to be used in a wide range of downstream tasks. Consequently, we consider EducOnto as successfully designed if it helps to fulfil downstream tasks related to high school and university information

retrieval. According to [37], it is then highly recommended to rely on a task-based evaluation as well.

To do so, we first take a general use case, after which we define several competency questions that encompass potential future applications. These competency questions will formalize what is expected from EducOnto. In a general use case, a high school senior may want to take advantage of the knowledge structured in EducOnto to retrieve suitable university curricula. An education professional could also use this knowledge to explore the curriculum catalog and extract relevant information. For instance, one may want to find out which university curriculum is the most popular for a given type of high school students. At this stage, we assume the ontology has been instantiated with real data. For more details about the building process of EduKG based on EducOnto, please refer to Section **??**.

In order to assess EducOnto's ability to address the potential applications it was designed for, we define the following competency questions:

CQ1: How to retrieve recommended/required academic backgrounds for a specific university curriculum?

CQ2: What is the most popular university curriculum for students coming from a given high school major?

CQ3: How to personalize university curricula recommendations based on a student profile?

CQ4: Based on some initial university curriculum a high school senior is interested in, what are the alternative curricula they might also consider?

- **CQ4.1**: What are the university curricula sharing the same keywords?
- **CQ4.2**: What are the university curricula related to the same field of study?

CQ5: How to mine the most influential factors leading to a particular university curriculum?

Many structural patterns prevail in the educational system. **CQ1** aims at retrieving the recommended or even required academic backgrounds for a specific university curriculum. For instance, a high school senior who is majoring in Literature may have difficulty in accessing scientific university curricula. As students may not be aware of all the restrictions that apply, this query returns all the university curricula that can be reasonably considered. However, some high school majors allow access to a large number of university curricula. In such circumstances, high school seniors may not only be interested in accessible university curricula. Instead, they may want to know what are the most picked-up curricula by high school seniors studying in the same major. **CQ2** answers this specific need. University curriculum recommendation should not be restricted to only factual information such as the high school major. **CQ3** evaluates the ability of EducOnto to return curricula that also match a student's personality and preferences. Keywords a student is interested in, or favorite school subjects are examples of such individual preferences. Besides, in some cases high school seniors already have some idea about what they would like to study at university. Nevertheless, they may not be aware of other similar curricula they could also consider. **CQ4** addresses this particular question by retrieving related curricula. Curricula can be interlinked on several criteria: common training type, keywords, field of study, or a combination of them. Finally, once high school seniors have chosen their university curriculum, designers of the ontology or education specialists might be interested in

taking a high-level perspective and identifying frequent patterns in students' profiles that lead them towards specific curricula. **CQ5** seeks to answer this need.

These competency questions can be addressed by querying EducOnto using SPARQL. The answers to the queries are retrieved from the data contained in EduKG. Below we present the results from the queries for **CQ1** and **CQ2**. For **CQ1**, we retrieve the recommended high school majors to start a Bachelor's degree in Computer Science (hereinafter referred to as *curriculum:lg_cs*). For **CQ2**, we retrieve the five favorite curricula for high school students majoring in sciences. The full list of queries and retrieved answers are available on the documentation page: https://nicolas-hbt.github.io/educ-ontokg/competencyquestions/.

```
PREFIX ed: <http://purl.org/educonto/>
PREFIX major: <http://purl.org/edukg/major/>
PREFIX curriculum: <http://purl.org/edukg/curriculum/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

SELECT ?major
WHERE {
    ?major rdf:type/rdfs:subClassOf* ed:HighSchoolMajor.
    curriculum:lg_cs ed:recommendsHighSchoolMajor ?major .
}
```

Listing 3.1: SPARQL query to address CQ1

```
PREFIX ed: <http://purl.org/educonto/>
PREFIX major: <http://purl.org/edukg/major/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

SELECT ?curriculum (COUNT(?curriculum) AS ?count)
WHERE
{ ?student rdf:type ed:UndergraduateStudent .
  ?student ed:likedCurriculum ?curriculum .
  ?student ed:pickedGeneralMajor major:ts .
}
GROUP BY ?curriculum
ORDER BY DESC(?count)
LIMIT 5
```

Listing 3.2: SPARQL query to address CQ2

3.3.2 EduKG

Data Collection

EduKG was built on the basis of data collected through a self-administered online survey, which comes in two versions: one for high school students and one for university students. The content of the survey has been developed in conjunction with education experts participating in the project AILES. Several high schools and universities have been asked to disseminate the survey among their students. We made sure to target a wide spectrum of geographical areas to ensure representativeness among our survey respondents. Alumni with no more than two years of work

experience were allowed to complete the survey as well. Their educational pathways provide high value patterns to learn from in order to make further recommendations to current students. The survey was made available online - thereby accessible to everyone. Some respondents graduated a long time ago and we assumed there may be some vagueness around their own past preferences as students. Besides, some university curricula labels and contents might have changed since then. So we choose to filter these respondents out to make sure not to introduce noise into the dataset. From January 7th 2022 to April 10th 2022, 3,583 respondents have been kept after the filtering process and are part of EduKG.

EduKG Construction and Description

Data were firstly collected in tabular format. Before constructing EduKG, feature engineering and selection were performed. Below, we specify some of our modelling choices.

- Students rated their curriculum on a 0 to 5 scale. Ratings were binarized so as to lead to the formation of two predicates: `educonto:likedCurriculum` (rating ≥ 3) and `educonto:dislikedCurriculum` (rating < 3);
- Students were allowed to provide keywords related to their interests. Because they were given a freeform text field to fill, consistency among respondents was needed. For instance, several students indicated their interest for "engineering", "engineering sciences" or "engineering projects". In that case, we standardized them all manually using the more general "engineering" keyword. Other standardization had to been made. By the same token, keywords that appeared too little were removed. Too precise keywords - often reflecting eclectic tastes - were replaced by a broader concept. For example, "Austrian economics" was replaced by "economics";
- Preliminary questions were meant to assess students' personality. They were asked to tick the boxes corresponding to statements that best describe them. Based on the amount of ticked boxes belonging to a defined archetype, students can be assigned a predominant personality trait. A similar process was applied to assess their academic skills;
- We do not limit ourselves to the data collected through the survey. Instead, we use publicly available datasets provided by government agencies to enrich the knowledge about the provision of university training. Additional annotations were provided by education experts.

Then we use Python 3.9 to create proper RDF subject-predicate-object triplets meeting the conditions specified by EducOnto. The set of all the defined triplets forms our knowledge graph EduKG.

Below we give examples about how a curriculum and a student are represented in Turtle syntax.

```
@prefix educonto: <http://purl.org/educonto/> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

<http://erpi.fr/educonto/curriculum/lg_cs> a educonto:Lg ;
    educonto:belongsToFieldOfStudy <http://erpi.fr/educonto/cs_mathematics> ;
    educonto:curriculumRelatesToKeyword <http://erpi.fr/educonto/abstraction>,
        <http://erpi.fr/educonto/algorithmique>,
        <http://erpi.fr/educonto/code>,
```

```
        <http://erpi.fr/educonto/informatique>,
        <http://erpi.fr/educonto/internet>,
        <http://erpi.fr/educonto/programmation>,
        <http://erpi.fr/educonto/web> ;
    educonto:recommendsHighSchoolMajor <http://erpi.fr/educonto/major/sti2d>,
        <http://erpi.fr/educonto/major/ts> ;
    educonto:recommendsHighSchoolSpecialty <http://erpi.fr/educonto/spe/math>,
        <http://erpi.fr/educonto/spe/nsi> .
```

Listing 3.3: RDF Specifications for the Bachelor's in Computer Science

```
@prefix educonto: <http://purl.org/educonto/> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

<http://erpi.fr/educonto/stud/user1302> a educonto:Person ;
    educonto:hasFavoriteSubject <http://erpi.fr/educonto/informatique>,
        <http://erpi.fr/educonto/mathematiques>,
        <http://erpi.fr/educonto/philosophie> ;
    educonto:hasPersonalityTrait <http://erpi.fr/educonto/investigative> ;
    educonto:hasSkill <http://erpi.fr/educonto/good_memory>,
        <http://erpi.fr/educonto/strong_confidence> ;
    educonto:likedCurriculum <http://erpi.fr/educonto/curriculum/lg_cs> ;
    educonto:pickedGeneralMajor <http://erpi.fr/educonto/major/ts> .
```

Listing 3.4: RDF Triplets for an undergraduate student in Computer Science

In Table 3.4, we provide a comprehensive view of EduKG dimensionality and degree of connectivity. Table 3.a counts the number of distinct entities for each intermediate class. Readers interested in using EduKG for making curriculum recommendations should take a closer look at Table 3.b, which reveals insightful information about the number of Student-Curriculum interactions and the relative sparsity of EduKG. The degree of sparsity regarding user-item interactions is extensively studied in the RS literature [95]. Table 3.c gives the total amount of entities, relations and triplets of EduKG. These statistics are helpful in assessing the relative size of EduKG. Inspired from [172, 187], we finally provide a general idea about how much the main entities are connected within EduKG (Table 3.5).

[16] Sparsity $= 1 - \frac{|R|}{|U| \times |I|}$, with $|I|$, $|R|$, and $|U|$ being the number of interactions, ratings, and users, respectively.

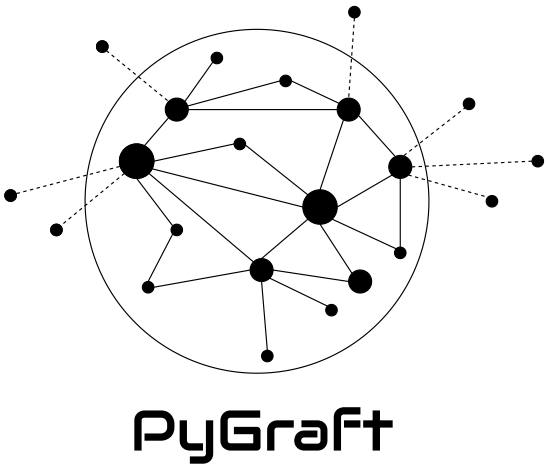
Table 3.4: Statistics of EduKG

Category	**Statistic**	**Value**
(a) Fields of Study	Number of Fields	12
	Number of Keywords	321
	Number of School Subjects	15
	Number of Majors	92
	Number of Specialties	13
(b) Users and Interactions	Number of Users (Students)	3,583
	Number of Items (Curricula)	286
	Number of Interactions	7,021
	Sparsity[16]	99.31%
(c) Knowledge Graph	Number of Entities	5,452
	Number of Relations	27
	Number of KG Triplets	36,301

Table 3.5: Number of inter-class relations in EduKG

Relation A-B	**#A**	**#B**	**#A-B**
Student-Subject	3,583	15	9,832
Student-Keyword	3,583	321	4,018
Curriculum-Keyword	286	321	718

3.4 Generating synthetic resources for knowledge-based applications



Following Section 3.3 that was a first attempt at providing semantically rich resources for KG-based applications, we extend this effort with the intent of providing a way for generating schema-augmented yet domain-agnostic resources. This is motivated by several factors: first, the process of brainstorming with domain experts and development for EducOnto, and the process of data collection, filtering, and processing for building EduKG are both labor-intensive and time-consuming. Second, this endeavour led us to propose one single ontology and KG, which remains insufficient for comprehensively evaluating the soundness of an approach, *i.e.* more resources of the same kind are needed. Third, the provided resources are highly specific, as they are restricted to the modelling of the transition between high school and university for curriculum recommendation. Consequently, at this point our main objective is to propose a way of generating resources that are domain-agnostic while still being underpinned by a schema and that can be used for the generation of several test beds, so that novel approaches can be benchmarked according to a variety of datasets.

To fulfill this, in this section we present PyGraft, an open-source Python-based tool that generates highly customized, domain-agnostic schemas and KGs. PyGraft seeks to address important gaps and shortcomings related to data availability, limited collection of benchmarks, and paucity of schema-based datasets for experimental purposes. More specifically, recent works outline that relying on a limited collection of datasets is not sufficient to assess the generalization capability of an approach. In some data-sensitive fields such as education or medicine, access to public datasets is even more limited. Ultimately, we mentioned in Section 3.2 the lack of schema-enriched datasets for some KG-based tasks. We propose PyGraft to remedy this.

3.4.1 Resource Description

Overview

From a high-level perspective, the entire PyGraft generation pipeline is depicted in Fig. 3.2. In particular, Class and Relation Generators are firstly initialized with user-specified parameters, and are then used for building the schema incrementally. The logical consistency of the schema

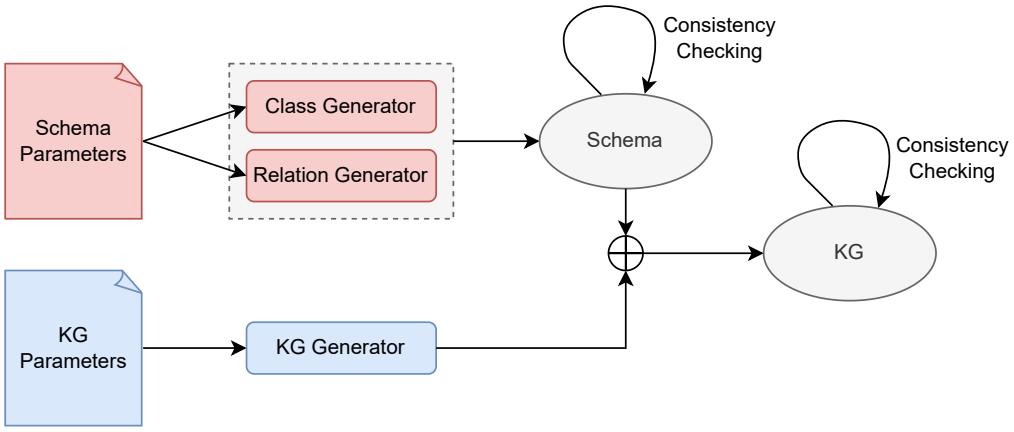


Figure 3.2: PyGraft general overview

is subsequently checked using HermiT reasoner [54] through the `owlready2`[17] Python library. If the user is also interested in generating a KG based on this schema, the KG Generator is initialized with KG-related parameters, and takes the previously generated schema as input in order to sequentially build the KG. Ultimately, the logical consistency of the resulting KG is assessed with HermiT. More details on the schema-level generation are provided in Section 3.4.1, while Section 3.4.1 describes the KG generation procedure.

Schema Generation

The schema generation follows a well-defined series of steps. In particular, the Class Generator in Fig. 3.2 is initialized first, and generates the class hierarchy and disjointness axioms as detailed in Algorithm 6. Then, the Relation Generator is initialized and handles the relation generation following the procedure of Algorithm 7. In the following, each algorithm is described step by step.

Class Generation. First, classes are generated based on the user-specified number of classes `num_classes` (lines 1-2). Then, the user-specified `max_depth` parameter is satisfied by taking one class after the other and creating child-parent connections through the `rdfs:subClassOf` assertion (lines 3-7). At this point, the class hierarchy is a purely vertical tree where each node (*i.e.* class) has exactly one child, except the leaf node. The class hierarchy is then further filled by taking each remaining class sequentially, and connecting it with other classes so that `avg_depth` and `inheritance_ratio` are satisfied (lines 8-13). It is worth mentioning that with some probability α chosen to be moderately low, a freshly picked class can be placed randomly (lines 10-11). This does not necessarily goes in the direction of `avg_depth` and `inheritance_ratio` target parameters, but it allows adding stochasticity and realism in the characteristics of the generated class hierarchy. Besides, when a low number of classes are still to be connected, this randomness is deactivated so that each subsequent class connection is in line with `avg_depth` and `inheritance_ratio` fulfilment. Finally, class disjointnesses are added to the schema by picking two classes A and B, ensuring that none of them is a transitive parent or child of the other, and extending class disjointness to their respective children, if any (lines 14-16).

When generating classes, anomalies may occur. For instance, choosing values such that `avg_depth` > `max_depth` triggers an error. In a few other situations, the schema is generated but

[17] `https://github.com/pwin/owlready2/`

Table 3.6: User-defined parameters for schema and KG generations

	Parameter	Description
Classes	`num_classes`	Number of classes
	`max_depth`	Depth of the class hierarchy
	`avg_depth`	Average class depth
	`inheritance_ratio`	Proportion of `rdfs:subClassOf`
	`avg_disjointness`	Proportion of `owl:DisjointWith`
Relations	`num_relations`	Number of relations
	`prop_profiled_relations`	Proportion of `rdfs:domain` and `rdfs:range`
	`relation_specificity`	Average depth of `rdfs:domain` and `rdfs:range`
	`prop_asymmetric`	Proportion of `owl:AsymmetricProperty`
	`prop_symmetric`	Proportion of `owl:SymmetricProperty`
	`prop_irreflexive`	Proportion of `owl:IrreflexiveProperty`
	`prop_reflexive`	Proportion of `owl:ReflexiveProperty`
	`prop_transitive`	Proportion of `owl:TransitiveProperty`
	`prop_functional`	Proportion of `owl:FunctionalProperty`
	`prop_inversefunctional`	Proportion of `owl:InverseFunctionalProperty`
	`prop_inverseof`	Proportion of `owl:inverseOf`
	`prop_subproperties`	Proportion of `rdfs:subPropertyOf`
Individuals	`num_entities`	Number of entities
	`num_triples`	Number of triples
	`relation_balance`	Relation distribution across triples
	`prop_untyped`	Proportion of untyped entities
	`avg_depth_specific`	Average depth of most specific class
	`multityping`	Whether entities are multi-typed
	`avg_multityping`	Average number of most-specific classes per entity

the user requirements might not be completely fulfilled. This can happen because of competing parameter values. For example, the constraints `num_classes` = 6 (`root` excluded), `max_depth` = 3, `avg_depth` = 1.5, and `inheritance_ratio` = 2.5 cannot be simultaneously satisfied (Fig. 3.3). In this situation, the schema is generated with a best-effort strategy, seeking to build a schema with statistics as close as possible to the user requirements.

Relation Generation. Before presenting the procedure for generating relations and their properties, it is worth mentioning that PyGraft allows relations to be described by multiple `OWL` and `RDFS` constructs. This leads to more realistic schemas and KGs, at the expense of higher risk of inconsistency: some property combinations are not logically consistent, *e.g.* a relation cannot be simultaneously qualified by `owl:ReflexiveProperty` and `owl:IrreflexiveProperty`. Based on the relation properties available in PyGraft (Table 3.7), all combinations were extracted and for each combination, a new graph was serialized using `rdflib`[18]. This graph contains a unique relation which is qualified by the given property combinations[19]. Based on the simplified schema, the HermiT reasoner performs consistency checking. Finally, a dictionary stores all possible property combinations. Knowing valid and invalid property combinations is necessary for guiding relation property assignment and minimize logical inconsistency likelihood before actually running the reasoner.

In Algorithm 7, this dictionary of valid property combinations is loaded (line 3) just after initializing the number of relations specified by the user (lines 1-2). Then, each relation in the schema is qualified with properties based on a pre-defined order (lines 4-6). For example, `owl:ReflexiveProperty` and `owl:IrreflexiveProperty` are assigned first, then `owl:SymmetricProperty`, etc. These properties are named *attributes* as they characterize a relation *per se*. Next, the *relationship property* `owl:InverseOf` is assigned to relations (line 7), which poses constraints on relation domain and range assignments (line 8), which themselves pose constraints on relation pairings through the `rdfs:subPropertyOf` assertion (line 9), *e.g.* domain and range of subproperties should not be disjoint with domain and range of superproperties.

In Table 3.7 are reported the relation properties that the current PyGraft version handles, along with their logical definition and accompanying examples.

Table 3.7: Relation properties covered by PyGraft

Property	Definition	Example
`owl:AsymmetricProperty`	$\forall x \forall y : p(x,y) \Rightarrow \neg p(y,x)$	`isParentOf`
`owl:SymmetricProperty`	$\forall x \forall y : p(x,y) \Rightarrow p(y,x)$	`hasSibling`
`owl:ReflexiveProperty`	$\forall x : p(x,x)$	`hasSameColorAs`
`owl:IrreflexiveProperty`	$\forall x : \neg p(x,x)$	`isYoungerThan`
`owl:TransitiveProperty`	$\forall x \forall y \forall z : p(x,y) \wedge p(y,z) \Rightarrow p(x,z)$	`isCheaperThan`
`owl:FunctionalProperty`	$\forall x \forall y \forall z : (p(x,y) \wedge p(x,z)) \Rightarrow y = z$	`hasISBN`
`owl:InverseFunctionalProperty`	$\forall x \forall y \forall z : (p(x,y) \wedge p(z,y)) \Rightarrow x = z$	`isEmailAddressOf`
`owl:InverseOf`	$\forall x \forall y : p(x,y) \Longleftrightarrow q(y,x)$	`owns` $\Longleftrightarrow$ `isOwnedBy`
`rdfs:subPropertyOf`	$\forall x \forall y : p(x,y) \Rightarrow q(x,y)$	`hasMother` $\Rightarrow$ `hasParent`

[18] `https://github.com/RDFLib/rdflib/`

[19] An instance triple should also be added. This is because some property combinations such as `owl:SymmetricProperty` and `owl:AsymmetricProperty` are not flagged as logically inconsistent *per se* in `OWL`. However, a relation qualified by these two properties is not allowed to connect any instances.

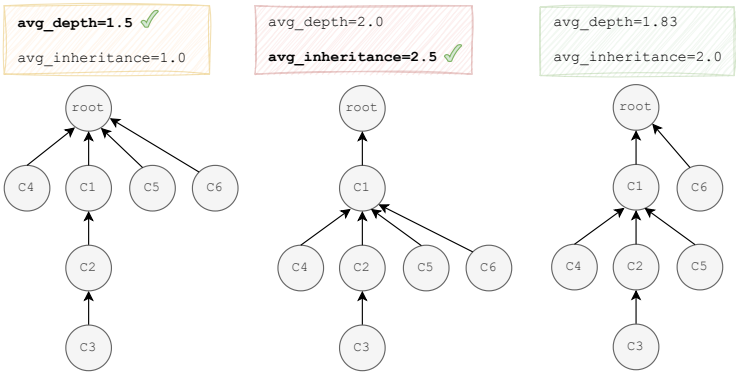


Figure 3.3: Potential class hierarchies for the constraints: `num_classes` = 6, `max_depth` = 3, `avg_depth` = 1.5, and `inheritance_ratio` = 2.5. Left and middle class hierarchies are built with parameter priority. The right class hierarchy is built with a best-effort strategy, without specific parameter privilege.

Knowledge Graph Generation

In light of PyGraft overview (Fig. 3.2), this section explores how the KG generator is initialized with user parameters and used in conjunction with any generated schema to build the final KG. As for schema, we provide a step-by-step insight into Algorithm 8. Entities and the KG are generated (lines 1-3) and a schema dedicated to the KG generation is loaded (line 4). Based on `prop_untyped`, entities are assigned a class whose depth in the class hierarchy should be in a close range around `avg_depth_specific` (line 5). Based on `avg_multityping`, several of them are assigned other classes of the same depth, provided that they are not disjoint with any of the specific classes characterizing a given entity (lines 6-7). Then, triples are generated (lines 8-9) in a sequential manner. Namely, unobserved entities in U have sampling priority, to ensure that the required number of entities in the resulting KG is met as fully as possible. Ultimately, checking procedures are performed (line 10) before the KG undergoes logical consistency checking using a semantic reasoner (line 11). If the generated KG is inconsistent, a message warns the user but the KG is stored nevertheless. The user can then choose to restart the generation procedure. It is worth mentioning that the checking procedures (line 10) follow the relevant `OWL 2 RL/RDF` rules[20]. These rules – in the form of first-order implications – are implemented in PyGraft to ensure consistency pre-checking before the HermiT reasoner is deployed. In addition, the OntOlogy Pitfall Scanner![21] was also used for identifying several checking procedures and for ensuring compliance with common Semantic Web standards.

3.4.2 PyGraft in Action

Efficiency and Scalability Details

In this section, the efficiency and scalability of PyGraft are benchmarked across several schema and graph configurations. Each schema specification reported in Table 3.8 is paired with each graph specification from Table 3.9. This leads to 27 distinct combinations.

In particular, schemas from $\mathcal{S}1$ to $\mathcal{S}3$ are small-sized, schemas from $\mathcal{S}4$ to $\mathcal{S}6$ are medium-sized, and schemas from $\mathcal{S}7$ to $\mathcal{S}9$ are of larger sizes (Table 3.8). For each schema of a given size,

[20] `https://www.w3.org/TR/owl2-profiles/#ref-owl-2-rdf-semantics/`
[21] `https://oops.linkeddata.es/`

the degree of constraints vary as they contain different levels of `OWL` and `RDFS` logical constructs. For example, $\mathcal{S}1$ has less constraints than $\mathcal{S}2$, which itself has less constraints than $\mathcal{S}3$. Graph specifications $\mathcal{G}_1$, $\mathcal{G}_2$, and $\mathcal{G}_3$ correspond to small-sized, medium-sized and large-sized graphs, respectively (Table 3.9).

For these 27 unique configurations, execution times w.r.t. several dimensions are computed and shown in Fig. 3.4. Execution times related to the schema generation are omitted as they are negligible. Experiments were conducted on a machine with 2 CPUs Intel Xeon E5-2650 v4, 12 cores/CPU, and 128GB RAM.

It is worth mentioning that the 27 generated KGs were flagged as consistent at the first attempt. The breakdown of time executions differs according to the schema and graph sizes (see Fig. 3.4). For small graphs, the final consistency checking is the most time-consuming part. For large graphs, triple generation time dominates the rest. As graph sizes increase, all execution times grow but we observe that PyGraft is able to generate consistent KGs quickly, even for large KGs: with our experimental configuration, the total execution time for KGs with 10K entities and 100K triples is roughly 1.5 minutes. In addition, PyGraft scalability was assessed by asking a KG of 100K entities and 1M triples. On the same machine, it took 47 minutes to generate such a KG, which was again consistent at the first attempt.

Table 3.8: Generated schemas. Column headers from left to right: number of classes, class hierarchy depth, average class depth, proportion of class disjointness (cd), number of relations, average depth of relation domains and ranges (rs), and proportions of reflexive (rf), irreflexive (irr), asymmetric (asy), symmetric (sy), transitive (tra), and inverse (inv) relations

	$\lvert\mathcal{C}\rvert$	MAX($\mathcal{D}$)	AVG($\mathcal{D}$)	cd	$\lvert\mathcal{R}\rvert$	rs	ref	irr	asy	sym	tra	inv
$\mathcal{S}1$	25	3	1.5	0.1	25	1.5	0.1	0.1	0.1	0.1	0.1	0.1
$\mathcal{S}2$	25	3	1.5	0.2	25	1.5	0.2	0.2	0.2	0.2	0.2	0.2
$\mathcal{S}3$	25	3	1.5	0.3	25	1.5	0.3	0.3	0.3	0.3	0.3	0.3
$\mathcal{S}4$	100	4	2.5	0.1	100	2.5	0.1	0.1	0.1	0.1	0.1	0.1
$\mathcal{S}5$	100	4	2.5	0.2	100	2.5	0.2	0.2	0.2	0.2	0.2	0.2
$\mathcal{S}6$	100	4	2.5	0.3	100	2.5	0.3	0.3	0.3	0.3	0.3	0.3
$\mathcal{S}7$	250	5	3.0	0.1	250	3.0	0.1	0.1	0.1	0.1	0.1	0.1
$\mathcal{S}8$	250	5	3.0	0.2	250	3.0	0.2	0.2	0.2	0.2	0.2	0.2
$\mathcal{S}9$	250	5	3.0	0.3	250	3.0	0.3	0.3	0.3	0.3	0.3	0.3

Table 3.9: Different graph specifications. Column headers from left to right: number of entities, number of triples, proportion of untyped entities, average depth of the most specific specific class, average number of most-specific classes per multi-typed entity

	$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{T}\rvert$	unt	asc	mul
$\mathcal{G}_1$	100	1,000	0.3	2.0	2.0
$\mathcal{G}_2$	1,000	10,000	0.3	2.0	2.0
$\mathcal{G}_3$	10,000	100,000	0.3	2.0	2.0

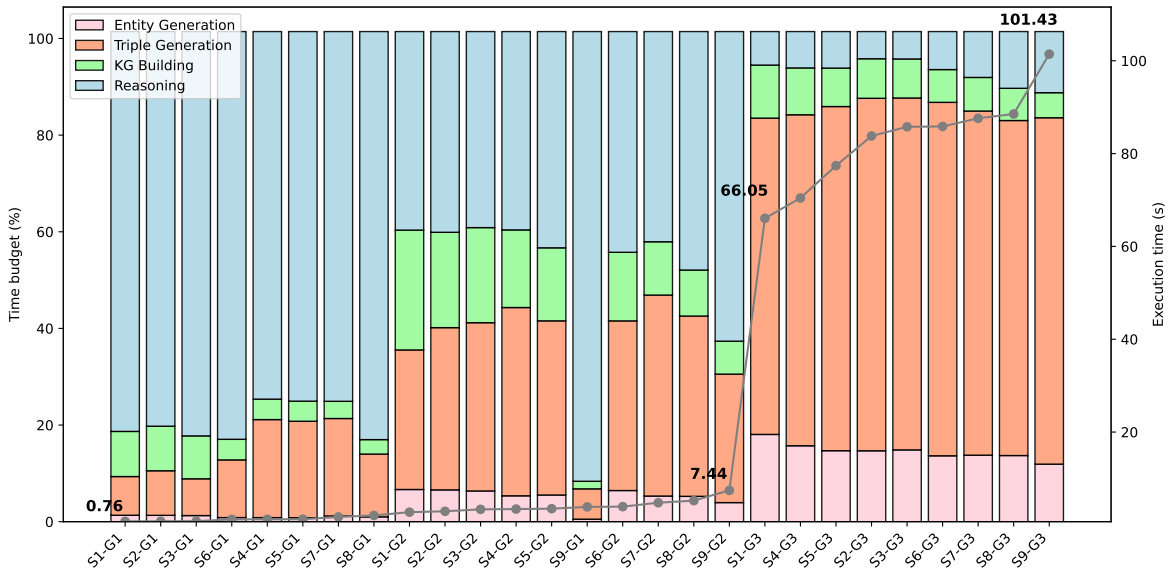


Figure 3.4: Execution time breakdown for each configuration

3.4.3 Usage Illustration

In this section, we briefly provide some usage examples to demonstrate how easy it is to use PyGraft. Desired characteristics of the output schema and/or KG can be specified with the path to a `yaml` or `json` configuration file. In the example presented in Listing 3.5, after importing PyGraft, we first generate a `yaml` configuration file in the current working directory. For the sake of simplicity, the generated template is left untouched, *i.e.* we keep the default parameter values. More advanced usage is provided in the official documentation: https://pygraft.readthedocs.io/en/latest/. Then, we generate both a schema and KG in a single pipeline. The generated resources are subsequently stored in the current working directory.

```
import pygraft

pygraft.create_yaml_template()
pygraft.generate("template.yml")
```

Listing 3.5: Schema and KG generation with PyGraft

By default, the generated graph is stored as an `rdf/xml` file. A snippet of a KG generated with PyGraft is provided in Listing 3.6.

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF
   xmlns:ns1="http://purl.org/dc/terms/"
   xmlns:owl="http://www.w3.org/2002/07/owl#"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   xmlns:sc="http://pygraf.t/"
>

  <rdf:Description rdf:about="http://pygraf.t/C30">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
```

```
  <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
  <owl:disjointWith rdf:resource="http://pygraf.t/C16"/>
</rdf:Description>
<rdf:Description rdf:about="http://pygraf.t/E324">
  <rdf:type rdf:resource="http://pygraf.t/C4"/>
  <rdf:type rdf:resource="http://pygraf.t/C17"/>
  <schema:R15 rdf:resource="http://pygraf.t/E356"/>
  <schema:R34 rdf:resource="http://pygraf.t/E44"/>
</rdf:Description>
<rdf:Description rdf:about="http://pygraf.t/R34">
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#TransitiveProperty"/>
  <rdfs:domain rdf:resource="http://pygraf.t/C17"/>
  <rdfs:range rdf:resource="http://pygraf.t/C17"/>
  <owl:inverseOf rdf:resource="http://pygraf.t/R15"/>
</rdf:Description>
```

Listing 3.6: Excerpt from a generated graph

3.5 Conclusion and future work

3.5.1 Recap

In this chapter, we have outlined a series of contributions that aim to enhance the resources available in the Semantic Web. This effort is crucial in facilitating the development of neuro-symbolic approaches across a range of applications that utilize knowledge graphs.

Initially, we identified a significant gap in the availability of datasets with accompanying schemas (*e.g.*, ontologies) in the specific area of link prediction. This lack of resources seems to impede the full potential of neuro-symbolic approaches. While it is feasible to manually query and enrich existing datasets, this method is inefficient and repetitive for each dataset. Besides, enriched datasets need to be shared so that such efforts are not repeated and that the community can benefit from these augmented resources for developing better approaches. To address this issue, we have extracted information regarding entity types, and the domains and ranges of relations from various knowledge graphs. We enriched mainstream datasets with such information and made the resulting datasets publicly available to foster neuro-symbolic initiatives around the task of link prediction.

The second contribution we presented is not concerned with a specific task such as link prediction, but focused on a specific domain of application: recommending university courses to high school students. Confronted with the absence of publicly available resources in this area, we created and shared EducOnto and EduKG -– an ontology and a knowledge graph for education. The process of designing questionnaires, collecting and preprocessing data, and building the ontology and knowledge graphs was extensive.

Reflecting on these efforts, we aimed to apply this approach more broadly, regardless of the application domain. Consequently, our final contribution lies in the development and release of PyGraft, a tool for generating custom, efficient, and adaptable synthetic schemas and knowledge graphs. PyGraft offers users significant control over resource configuration. We hope that PyGraft will be used to diversify benchmark datasets, foster new approaches, and more generally support the development of neuro-symbolic methods in the field of knowledge graphs.

3.5.2 Future work

The contributions presented in this Chapter can be further enhanced in many ways. Our contribution related to EducOnto and EduKG can be strengthen by working on the following research directions:

Mapping to external knowledge bases. We would like to enrich EduKG by mapping its entities to DBpedia or Wikidata entities. This would give more genericity to EduKG and would facilitate its use in downstream tasks by leveraging the wealth of information which is already available in the SW ecosystem.

SPARQL endpoint. We also intend to provide a public SPARQL endpoint, so that information can be retrieved by running SPARQL queries against EduKG.

Data collection. We should first recall that data about university education were collected in a specific country. Although most of the university curricula contained in EduKG are generic enough to hold regardless of the educational system at hand, the titles of some other curricula may not have a direct translation into another educational setting. We consequently intend to broaden EduKG's scope by adding instances from other European countries, although its feasability will depend on privacy-related and data-sharing concerns.

Recommender system. Finally, recall that the ultimate goal for building EduKG was to provide a public dataset for curriculum recommendation. Thus, the natural next step of this work is to design a RS that specifically relies on EduKG.

In addition, we should also point out future research directions to improve PyGraft:

Increasing the size of generated resources. It should be noted that PyGraft currently relies on `rdflib` for serializing triples. We tried to push PyGraft capabilities to its limit by generating very large KGs with >10M entities and triples. In such cases, the serialization failed. This is why, in future work, we will aim at developing independent serialization procedures so that bigger graphs can be generated.

Handling specific concurrent properties. The current PyGraft version cannot fully handle the following properties at the same time: `rdfs:subPropertyOf`, `owl:FunctionalProperty`, and `owl:InverseFunctionalProperty`. When the three of them have a non-null value, the generated KGs is more prone to being inconsistent. However, in our experiments, this issue mainly happens when asking for large to very large graph sizes. In future work, we will extend the set of checking procedures to better encompass any sources of inconsistency.

Improved consistency checking procedure. Our implemented checking procedures limit the likelihood of any inconsistency *before* the DL reasoner is ultimately applied on the generated KG. In case of inconsistencies that would remain undetected by the checking procedures, our actual use of the HermiT reasoner is able to detect such inconsistencies, but not to provide any information on the triples that should be removed so that the KG becomes consistent. In future work, we will implement such a functionality, so that the consistency of the generated KGs can be ensured in a single loop without requiring any input from the user's side.

Chapter 4

Evaluating the semantic awareness of knowledge graph embedding models

Contents

This chapter covers the following articles:

Sem@K: Is my knowledge graph embedding model semantic-aware? Hubert *et al.* (2023). Semantic Web – Interoperability, Usability, Applicability, December 2023, 14 (6), pp.1273-1309, DOI:10.3233/SW-233508.

Knowledge Graph Embeddings for Link Prediction: Beware of Semantics! Hubert *et al.* (2022). Proceedings of the Workshop on Deep Learning for Knowledge Graphs (DL4KG 2022) co-located with the 21th International Semantic Web Conference (ISWC 2022), Virtual Conference, October 24, 2022.

New Strategies for Learning Knowledge Graph Embeddings: the Recommendation Case. Hubert *et al.* (2022). Knowledge Engineering and Knowledge Management - 23rd International Conference, EKAW 2022, Bolzano, Italy, September 26-29, 2022. (**Best Paper Award**)

4.1 Motivations and contributions

This preliminary section aims at going beyond the pure description of rank-based metrics introduced in Section 2.3.5, by providing an in-depth analysis of their characteristics. In particular, we show that their use as the only criterion for assessing KGEM accuracy reveals a performance stagnation in newly proposed models (Section 4.1.1). This raises the concern of whether other metrics can be used to differentiate models of seemingly equivalent quality. This concern becomes even more crucial when taking into account the intrinsic flaws and biases of rank-based metrics, which ultimately calls for a more holistic evaluation framework for KGEMs (Section 4.1.2).

4.1.1 Marginal gains and implementation disparities

The number of KGEMs proposed in the last decade is enormous, and new KGEMs are still being introduced in recent works [170, 28]. However, the performance gains achieved on popular benchmarks have been stalling for a few years: an excerpt from Papers With Code (https://paperswithcode.com/) depicted in Fig. 4.1 clearly shows that latest KGEMs do not offer any performance boost over NBFNet [190] – the best model presented in 2021.

In addition to the diversity in proposed KGEMs and the observed stagnation in performance, we cannot neglect the variety of open-source libraries implementing these models. Such libraries are commonly used by researchers to train and evaluate their approaches with off-the-shelf models, negative samplers, and evaluation pipelines. The handful of readily available libraries is a boon for the visibility and development of methods based on KGEMs. However, to quote [3]: "The heterogeneity in recently published knowledge graph embedding models' implementations, training, and evaluation has made fair and thorough comparisons difficult". Table 4.1 reports the link prediction results of TransE on FB15k237 with different popular libraries. Notably, we clearly see that a well-studied KGEM such as TransE can exhibit a high variability in its performance based on the chosen implementation. If new KGEMs continue being introduced with little to no difference in terms of rank-based metrics' performance, one might legitimately

Table 4.1: TransE link prediction results on FB15K237. * denotes the results obtained in [184] (NeuralKG) using the library under consideration, while $^{\Delta}$ denotes the results reported by the authors of the library. As an example, DGL-KE[189]* means that the results shown in the Table are the ones provided in [184] (NeuralKG).

Library	MRR	Hits@1	Hits@3	Hits@10
AmpliGraph[34]$^{\Delta}$	0.31	0.22	0.35	0.50
DGL-KE[189]*	0.23	0.13	0.27	0.43
LibKGE[22]$^{\Delta}$	0.31	0.22	0.35	0.50
NeuralKG[184]	0.32	0.23	0.36	0.51
OpenKE[63]	0.29$^{\Delta}$	0.19*	0.32*	0.48$^{\Delta}$
PyKEEN 1.0[4]*	0.14	0.05	0.11	0.30
Pykg2vec[182]*	0.25	0.16	0.28	0.42
TorchKGE[20]*	0.29	0.20	0.31	0.46

wonder whether additional metrics could be used to differentiate them on the basis of other characteristics. Relying on the sole use of rank-based metrics, and in the absence of significance test, the conclusion would simply be that a handful of KGEMs perform equally, and that they can be used interchangeably in production environment. Obviously, drawing such conclusion is undesirable. In this work, we posit that these KGEMs cannot be used interchangeably, and that their differences could become clear if we study them under dimensions that differ from rank-based metrics.

4.1.2 Addressing the limits of rank-based metrics

Beyond the performance stagnation of state-of-the-art (SOTA) model performance w.r.t. rank-based metrics and the implementation disparities leading to divergent results (Section 4.1.1), a critical look at rank-based metrics themselves has been undertaken in recent works. In what follows, we highlight the limits of such metrics, as well as recent works that address them.

Bias sensitiveness. LP is often used to complete knowledge graphs, where the Open World Assumption (OWA) prevails. KGs are incomplete and, due to the OWA, an unobserved triple used as a negative one can still be positive. It follows that traditional evaluation methods based on rank-based metrics may systematically underestimate the true performance of a KGEM [173].

In addition, Mohamed *et al.* [113] show that the traditional rank-based metrics for LP – Hits@K, MR, and MRR – are biased towards popular entities and relations, which appear in many triples. More precisely, they demonstrate that these metrics provide a biased estimation of KGEM performance as they under-emphasize the prediction's value of long-tail entities and relations, so that relying solely on these metrics inevitably leads to developing models that perform well on popular entities and relations but ignore the rest. The authors consequently propose two new, corrected rank-based metrics – Strat-Hits@K and Strat-MRR – which are unbiased estimators of Hits@K and MRR, respectively.

Intrinsic shortcomings. Hits@K, MR, and MRR have are intrinsically flawed, as pointed out in several works [10, 70, 162]. For example, Hits@K does not take into account triples whose rank is larger than K. As such, a model scoring the ground-truth in position $K+1$ would be considered equally good as another model scoring the ground-truth in position $K+d$ with $d \gg 1$. It follows that Hits@K is not a suitable metric for drawing comparisons between

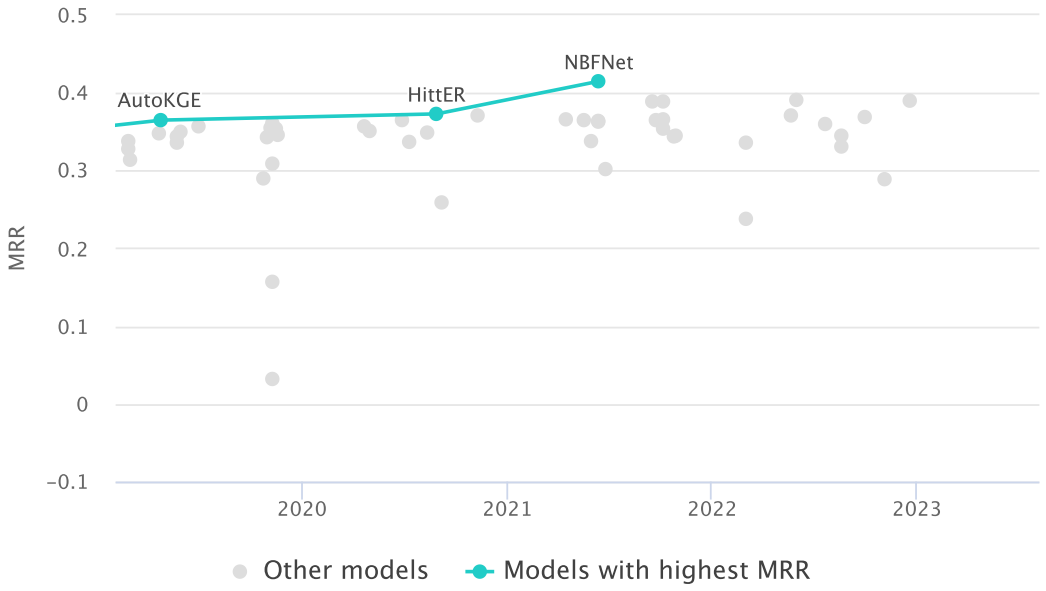


Figure 4.1: Evolution of link prediction results (MRR) on FB15k237

models [70]. MR alleviates this concern as it does not consider any threshold K. Therefore, MR allows to compare KGEM performance on the same dataset. Nonetheless, MR is sensitive to the number of KG entities (see Eq. (2.12)) [10]: a MR of 10 indicates very good performance if the set of entities is in the thousands, but it would indicate poor performance if the set of entities is much more restricted. Therefore, MR does not allow comparisons across datasets. Recent works recommend using adjusted version of the aforementioned metrics. The Adjusted Mean Rank (AMR) proposed in [3] compares the mean rank against the expected mean rank under a model with random scores. In [10], Berrendorf *et al.* transform the AMR to define the Adjusted Mean Rank Index (AMRI) bounded in the $[0, 1]$ interval. This way, a value of 1 indicates an optimal performance of the model. A value of 0 indicates a model performance similar to a model assigning random scores. A negative value indicates that the model performs worse than random. However, all these attempts at producing a better evaluation framework still focus on the quantitative assessment of KGEMs, *i.e.* the improvement of already existing rank-based metrics.

Single-faceted evaluation. As noted in Section 4.1.1, performance of newly proposed KGEMs on popular LP benchmarks are stagnating. In particular, Fig. 4.1 displays performance of schema-agnostic KGEMs, *i.e.* models that learn to predict links in the KG solely on the basis of relational triples between individuals. These are pure ML-based models that operate without any additional information being injected during training. Popular models such as TransE, ComplEx, and ConvE fall under this category. This is why Chapter 2 is dedicated to them. However, in some cases, the underlying semantics of KGs is considered as an additional source of information during training [111, 35, 87]. However, these attempts are intended to improve KGEM performance in terms of Hits@K, MR, and MRR. The ability of KGEMs to generate predictions in accordance with these semantic constraints is never directly addressed. This encourages further assessment of the semantic capabilities of KGEMs, as firstly suggested in [75]. In this chapter, we directly address this issue by assessing to what extent KGEMs are able to give high scores to triples

whose head (resp. tail) belongs to the domain (resp. range) of the relation.

Towards more comprehensive evaluation frameworks. As pointed out above, the traditional evaluation of KGEMs solely based on rank-based metrics can be flawed for several reasons. First, rank-based metrics are prone to reflect and even amplify the popularity biases present in mainstream benchmarks [113]. Second, they have their own intrinsic shortcomings, which calls for increased cautiousness when drawing conclusions on their sole basis. Third, even when their shortcomings would be alleviated by using their unbiased [113] and adjusted [3, 10] versions, they are still inherently concerned with one single aspect of KGEM evaluation. This can cause different KGEMs benchmarked on the same test sets to exhibit very similar results (Section 4.1.1). Using only rank-based metrics, a flawed conclusion would be that a few KGEMs are best-performers, and that they can be used interchangeably. Even when slight differences would be noticeable, the final choice of model would depend on the best achieved MRR and/or Hits@K. This raises the questions whether the chosen model is actually the best one, or whether its relative superiority over other KGEMs can be due to other factors such as better hyperparameter tuning or better modeling of a relational pattern highly present in the test set. In this chapter, we posit that using only rank-based metrics does not provide the full picture of KGEM quality for the downstream LP task, as some dimensions of KGEMs are left unassessed. To motivate the need for a broader evaluation framework that would, for instance, encompasses a semantic aspect, this section builds upon a minimalist example which is representative of the issue encountered while benchmarking the performance of several KGEM on the same dataset. As depicted in Fig. 4.2, two KGEMs that have been trained on the whole training set are tested on a batch of test triples. These KGEMs are referred to as Model A and Model B. Without loss of generality, it is assumed that the test set only comprises the three triples shown in Fig. 4.2. For the sake of clarity, only the tail prediction pass and the top-5 ranked candidate entities are depicted in Fig. 4.2. It should be noted that the performance of both models are strictly equal in terms of MR, MRR and Hits@K with $K \leq 5$: MR=8/3, Hits@1=1/3, Hits@3=2/3 and Hits@5=3/3. MRR can only be computed knowing the total number of entities in the KG but two models having the same MR on the same dataset have *de facto* the same MRR as well. Distinguishing between these two models by relying solely on traditional rank-based metrics is not possible. One might even draw the misleading conclusion that these models are equally good. But they are not: Model A gives high scores to semantically valid entities with regard to the range of relations (*e.g.*, for each prediction, the best-ranked entity is semantically valid), while Model B semantic awareness is very low (*e.g.*, for the first prediction, none of the top-3 ranked entities is semantically valid). In other words, Model B does not capture the semantic profile of relations well. This case in point illustrates the need for additional metrics to better assess the overall quality of KGEMs. In this thesis, we focus on semantic-oriented metrics.

The above discussion makes clear that relying on rank-based metrics only is not desirable. As such, this chapter is dedicated to answering our second research question:

> **RQ2.** Given the sole reliance on rank-based metrics for evaluating KGEMs, can we propose new metrics that would qualitatively assess different aspects of these models, especially their semantic awareness?

In this chapter, we more specifically focus on the study of the semantic awareness of KGEMs by proposing a novel metric named Sem@K. The semantic awareness of a wider range of KGEMs is analyzed across several datasets, in order to fairly assess the semantic capabilities of KGEMs. Thus, the following research questions are addressed:

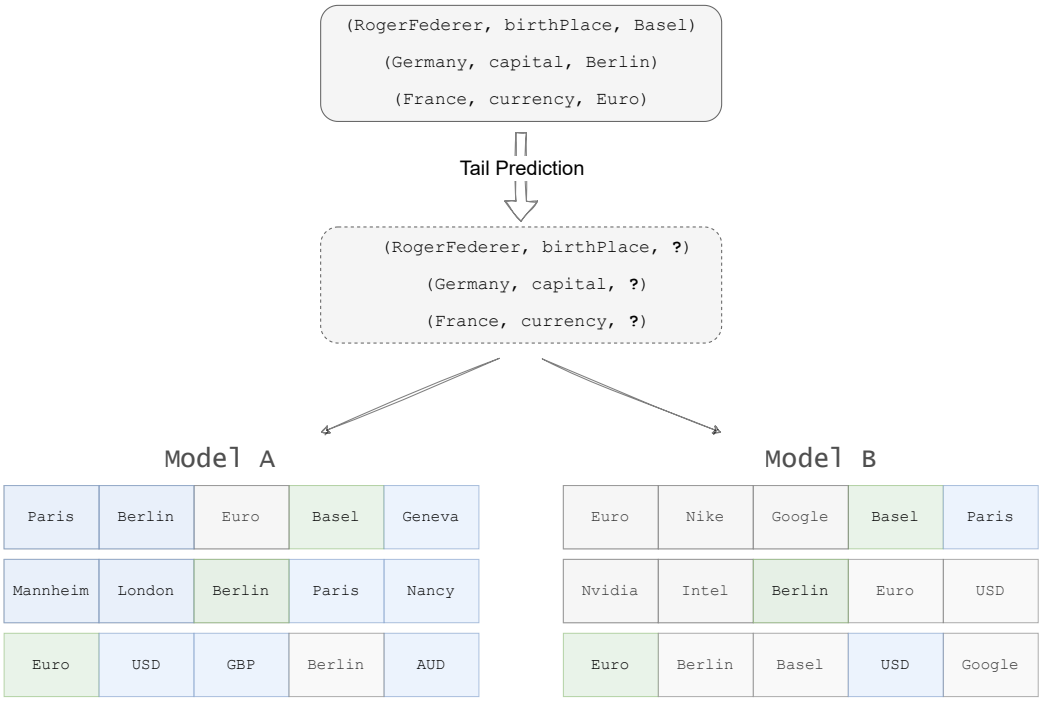


Figure 4.2: Motivating example. Tail prediction is performed for the three test triples contained in the upper insert. Model A and Model B output scores for each possible entity and only the top-five ranked tail candidates are depicted here. Model A and Model B have the same Hits@1, Hits@3 and Hits@5 values. But Model A has better semantic capabilities. Green, blue and white cells respectively denote the ground-truth entity, entities other than the ground-truth and semantically valid, and entities other than the ground-truth and semantically invalid.

RQ2.1. How the evaluation of KGEM semantic awareness should adapt to the typology of KGs?

RQ2.2. How semantic-aware agnostic KGEMs are?

RQ2.3. Does the evaluation of KGEM semantic awareness offer different conclusions on the relative superiority of some KGEMs compared to rank-based metrics?

RQ2.4. How can Sem@K be used in real-world downstream tasks, *e.g.* recommendation?

Accordingly, the main contributions of this chapter are:

1. To evaluate KGEM semantic awareness we propose Sem@K, a new semantic-oriented metric that comes in different versions so as to support a broad range of KGs (Section 4.2) (**RQ2.1**).

2. Our study supports the view that agnostic KGEMs are quickly able to infer the semantics of KG entities and relations (Section 4.3) (**RQ2.2**).

3. We consequently perform an extensive study of the semantic awareness of state-of-the-art KGEMs on mainstream KGs as well as a dynamic study of the evolution of KGEM semantic awareness vs. their performance in terms of rank-based metrics along training epochs (Section 4.3) (**RQ2.3**).

4. We give a concrete example of why Sem@K matters in downstream tasks such as recommendation, and how it can be used to measure the semantic validity of the top-ranked candidate items (Section 4.4) (**RQ2.4**).

4.2 Sem@K: a novel semantic-oriented metrics

The standard LP evaluation protocol consists in reporting aggregated results, considering the aforementioned rank-based metrics. As discussed in Section 4.1.2, these metrics only provide a partial picture of KGEM performance. To give a more comprehensive assessment of KGEMs, we aim at assessing their semantic awareness using our proposed metric called Sem@K that we discuss below. In particular, Sem@K is a general-purpose metric that can be instantiated with many variants. After motivating the use of semantic-oriented metrics (Fig. 4.2), we present 3 different versions of Sem@K: Sem@K(base) (Section 4.2.2), Sem@K(ext) (Section 4.2.4), and Sem@K(wup) (Section 4.2.5). Importantly, these different flavors of Sem@K are appropriate for different typologies of KGs, which are summarized in Table 4.2 and further detailed below. In the following, when no suffix is provided, it is assumed that we are concerned with Sem@K in general, regardless of the actual version.

4.2.1 Semantic Validity

We start by briefly defining what we mean by "semantic validity". Let us assume we are given a ground-truth triple $q = (h, r, t)$, and $\mathcal{S}_q^K$ is the list of the top-K candidate triples scored by a KGEM (*i.e.* by predicting the tail for $(h, r, ?)$ or the head for $(?, r, t)$). In this work, by semantic compatibility we refer to the fact that the predicted head (resp. tail) belongs to the domain (resp. range) of the relation. We will stick to this definition in the rest of this manuscript.

Table 4.2: Typology of KGs and their respective adequacy for the presented Sem@K versions.

KG type	Sem@K(ext)	Sem@K(base)	Sem@K(wup)
Schemaless	×		
Schema-defined, w/o class hierarchy	×	×	
Schema-defined, w/ class hierarchy	×	×	×

4.2.2 Definition of Sem@K(base)

This version of Sem@K (Eq. (4.1)) accounts for the proportion of triples that are semantically valid in the first K top-scored triples:

$$\text{Sem@}K = \frac{1}{|\mathcal{T}|} \sum_{q \in \mathcal{T}} \frac{1}{K} \sum_{q' \in \mathcal{S}_q^K} \text{compatibility}(q, q') \tag{4.1}$$

The function compatibility(q, q') (Eq. (4.2)) assesses whether the candidate triple q' is semantically compatible with its ground-truth counterpart q, considering our definition of semantic validity as defined in Section 4.2.1:

$$\text{compatibility}(q, q') = \begin{cases} 1, & \text{if } \text{types}(q'_h) \cap \text{domain}(q_r) \neq \emptyset \wedge \text{types}(q'_t) \cap \text{range}(q_r) \neq \emptyset \\ 0, & \text{otherwise} \end{cases} \tag{4.2}$$

where types(e) returns the types of entity e and domain(r) (resp. range(r)) is the domain (resp. range) of the relation r. q_r, q'_h, and q'_t denote the ground-truth relation, the head and the tail of the ranked triple q', respectively. It is noteworthy that with this formula, we allow entities to instantiate multiple types, and domains and ranges to be defined with multiple types. Sem@K is bounded in the $[0, 1]$ interval. Compared to Hits@K (Eq. (2.11)), Sem@K is non-monotonic: increasing K can lead to either lower or higher Sem@K values.

4.2.3 A note on Sem@K, untyped entities, and the OWA

Traditional KGEM evaluation can be performed in all situations, regardless of whether entities are typed or whether the KG comes with a proper schema. When measuring the semantic capability of KGEMs – *e.g.* with Sem@K – some concerns arise. For instance, a fair question to ask is the following: how should untyped entities be considered? Indeed, in some KGs, some entities are left untyped. For example, in DBpedia the entity `dbr:1._FC_Union_Berlin_players` does not belong to any other class than `owl:Thing`. In some other cases as in DB15K and DB100K [35], entities have incomplete typing.

Under the OWA, an untyped entity should not count in the calculation of Sem@K since it is not possible to determine whether this entity has no types or no known types. Although this seems to be a fair option, it raises a major issue: it makes it possible to score different sets of entities with rank-based metrics and Sem@K, which is not desirable. If there are M untyped entities in an ordered list of ranked entities, Hits@K, MR and MRR are calculated regardless of this fact, *i.e.* still taking into account the M untyped entities. However, Sem@K cannot be calculated for these M untyped entities. As such, MR, MRR and Hits@K would be calculated on the original entity set, whereas Sem@K would be computed on a different set of entities. This issue would be even more acute in the case of Sem@1 when the first ranked entity is untyped: Hits@1 and Sem@1 would be calculated on two different entities, which is not

acceptable. Consequently, one strategy consists in removing untyped entities from the evaluation protocol, both regarding rank-based and semantic-based metrics. By doing so, consistency is ensured in the ranked list of entities across rank-based and semantic metrics evaluation. However, one potential downside of this approach is that long-tail entities for which we have less information will not be considered, which might entail a bias towards popular entities, for which we have much more information.

4.2.4 Sem@K(ext) for schemaless KGs

Not all KGs come with a proper schema, *e.g.* relations do not appear in any `rdfs:domain` or `rdfs:range` clauses. In that particular situation, it can still be desirable to assess the semantic awareness of KGEMs. One approach is to maintain a list of all entities that have been observed as heads (resp. tails) of each relation r : $\text{domain}(r) = \{e : \exists(e, r, t) \in \mathcal{T}\}$ (resp. $\text{range}(r) = \{e : \exists(h, r, e) \in \mathcal{T}\}$). Hence, in this case, Sem@K(ext) is defined as in Eq. (4.1) and (4.2) but using these definitions of domain and range. Therefore, an entity will be considered as a semantically valid head (resp. tail) with respect to the relation if this entity appears as a head (resp. tail) in any other triple observed in the KG with the same relation. However, this approach is implicitly biased towards observed data, as it inherently assumes that the existing data is a comprehensive representation of all semantically valid triples. This bias can overlook the possibility of valid but unobserved triples, potentially penalizing correct inferences that fall outside the observed KG. Besides, schemaless KGs can contain errors or noisy data, including incorrect or misleading relations between entities. Basing domain and range definitions on such data can propagate these inaccuracies through the semantic validity assessments, affecting the reliability of the evaluation. Hard-coded domains and ranges are more reliable information. For this reason, Sem@K(ext) should only be used when Sem@K(base) cannot be considered.

4.2.5 Sem@K(wup): hierarchical Sem@K based on Wu-Palmer similarity

Sem@K as previously defined equally penalizes all entities that are not of the expected type. However, KGs may be equipped with a class hierarchy that, in turn, can support a more fine-grained penalty for entities depending on the distance or similarity between their type and the expected domain (resp. range) in this hierarchy. To illustrate, consider Figure 4.3 that depicts a subset of the DBpedia ontology `dbo` class hierarchy. Using the hierarchy-free version of Sem@K for the test triple (`dbr:The_Social_Network`, `dbo:director`, `dbr:David_Fincher`), predicting `dbr:Friends` or `dbr:Central_Park` as head would be penalized the same way in the compatibility function. However, it is clear that an entity of class `dbo:TelevisionShow` is semantically closer to `dbo:Film` – the domain of the relation `dbo:director` – than an entity of class `dbo:Park`, and thus should be less penalized.

To leverage such a semantic relatedness between concepts in Sem@K, the compatibility function can be adapted accordingly:

$$\text{compatibility}(q, q') = \min \begin{pmatrix} \max\limits_{\substack{c \in \text{type}(q'_h) \\ c' \in \text{domain}(q_r)}} \sigma(c, c'), & \max\limits_{\substack{c \in \text{type}(q'_t) \\ c' \in \text{range}(q_r)}} \sigma(c, c') \end{pmatrix} \tag{4.3}$$

where $\sigma(c, c')$ measure the semantic similarity between the two classes c and c' based on the class hierarchy. It should be noted that in this formula type(e) only returns the most specific classes instantiated by e.

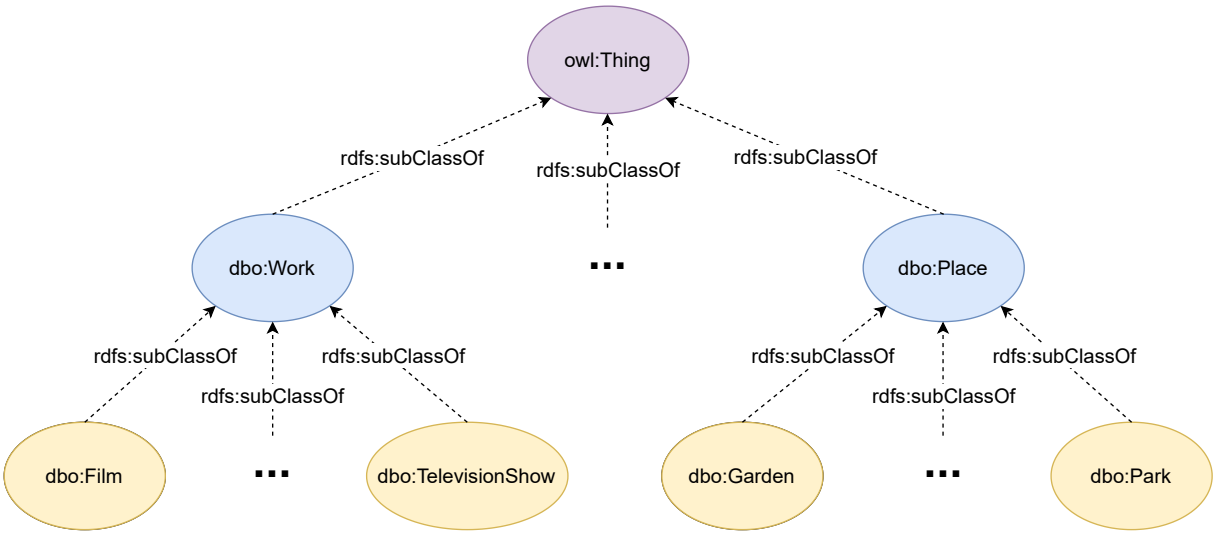


Figure 4.3: Excerpt from the DBpedia class hierarchy

Several similarity measures σ have been proposed in the literature [140, 177, 100, 142, 102]. In this work, the Wu-Palmer similarity [177] (Eq. 4.4) is used:

$$\sigma\left(c, c'\right) = \frac{2 \times \delta\left(c \wedge c', \rho\right)}{\delta\left(c, c \wedge c'\right) + \delta\left(c', c \wedge c'\right) + 2 \times \delta\left(c \wedge c', \rho\right)} \tag{4.4}$$

where ρ is the root of the hierarchy (*e.g.* `owl:Thing`), $\delta\left(c, c'\right)$ is the number of edges linking c to c', and $c \wedge c'$ represents the least common subsumer of c and c'. The Wu-Palmer similarity is well suited to a class hierarchy and provides a good indication of the semantic relatedness between the domain (resp. range) class and the classes of the chosen entity. This gives rise to the Sem@K(wup) version, where "wup" denotes the Wu-Palmer similarity. Note that other choices of similarity metrics based on class hierarchy are possible. In such cases, Sem@K(wup) should be renamed after the chosen simialrity metric.

Considering the example in Fig. 4.3, using the Wu-Palmer score in the calculation of Sem@K, a head prediction of `dbr:Friends` and `dbr:Central_Park` for the ground-truth triple (`dbr:The_Social_Network`, `dbo:director`, `dbr:David_Fincher`) are now differently penalized. The instance `dbr:Friends` is of type `dbo:TelevisionShow`, so we have: σ (`dbo:TelevisionShow`, `dbo:Film`) = 1/2. The instance `dbr:Central_Park` is of type `dbo:Park`, so we have: σ (`dbo:Park`, `dbo:Film`) = 0. This illustrates that incorporating the Wu-Palmer score into Sem@K calculation leads to more precise semantic comparisons that take into account the available class hierarchy. It should be noted that comparing the classes of a candidate entity with the expected class can result in the same penalization as the base version of Sem@K. However, in most cases, two classes do not lie in a completely disjoint part of the hierarchy of classes. As such, the Wu-Palmer score between the classes of a candidate entity and the expected class is rarely 0.

4.3 A comprehensive analysis of the semantic awareness of knowledge graph embedding models

4.3.1 Experimental setting

Datasets. In order to draw reliable and general conclusions, a broad range of datasets are used in our experiments. They have been chosen due to their different characteristics (*e.g.* entities, relations, classes, presence of a class hierarchy). In particular, 4 schema-enriched and 3 schemaless datasets are used in this section. Their descriptions and statistics are provided in Sections 3.2.2 and 2.5. We recall their statistics in Tables 4.3 and 4.4.

Table 4.3: Statistics of the schema-defined KGs

Dataset	$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{R}\rvert$	$\lvert\mathcal{C}\rvert$	$\lvert\mathcal{T}_{train}\rvert$	$\lvert\mathcal{T}_{valid}\rvert$	$\lvert\mathcal{T}_{test}\rvert$
DB93k	92,574	277	311	237,062	18,059	36,424
FB15k237-ET	14,541	237	643	272,115	17,535	20,466
YAGO3-37k	37,335	33	132	351,599	4,220	4,016
YAGO4-19k	18,960	74	1,232	27,447	485	463

Table 4.4: Statistics of the schemaless KGs

Dataset	$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{R}\rvert$	$\lvert\mathcal{T}_{train}\rvert$	$\lvert\mathcal{T}_{valid}\rvert$	$\lvert\mathcal{T}_{test}\rvert$
CoDEx-S	2,034	42	32,888	1,827	1,827
CoDEx-M	17,050	51	185,584	10,310	10,311
WN18RR	40,943	11	86,834	3,034	3,134

Baseline models. In this section, the semantic awareness of the most popular semantically agnostic KGEMs is analyzed. More specifically, the translational models TransE [18] and TransH [174], the semantic matching models DistMult [181], ComplEx [164] and SimplE [93], and the convolutional models ConvE [39], ConvKB [120], R-GCN [147] and CompGCN [166] are considered. Note that in the analysis of the results in Section **??**, a distinction will be made between pure convolutional KGEMs (ConvE, ConvKB) and GNNs (R-GCN, CompGCN). The characteristics of the models used in our experiments are fully described in Section 2.4 and briefly summarized in Table 4.5.

Implementation and hyperparameters. For the sake of comparisons, MRR, Hits@K and Sem@K all need to rely on the same code implementation. More specifically, for R-GCN[22] and CompGCN[23], existing implementations were reused and Sem@K values were calculated on the trained models. Other KGEMs were implemented in PyTorch. To avoid time-consuming hyperparameter tuning, we took inspiration from the hyperparameters provided by LibKGE [22] for CoDEx-S, CoDEx-M, WN18RR and FB15k237-ET. However, LibKGE does not benchmark all the datasets and models used in our experiments. For such models, we stick to the hyperparam-

[22] `https://github.com/tooooooodo/RGCN-LinkPrediction`
[23] `https://github.com/malllabiisc/CompGCN`

Table 4.5: KGEMs used in the experiments

Model Family	Model	Scoring Function	Loss Function
Geometric	TransE	$\|\mathbf{e}_h + \mathbf{e}_r - \mathbf{e}_t\|_p$	Pairwise Hinge
	TransH	$\|\mathbf{e}_{h_\perp} + \mathbf{d}_r - \mathbf{e}_{t_\perp}\|_p$	Pairwise Hinge
Semantic Matching	DistMult	$\langle \mathbf{e}_h, \mathbf{W}_r, \mathbf{e}_t \rangle$	Pairwise Hinge
	ComplEx	$\mathrm{Re}\left(\mathbf{e}_h \odot \mathbf{e}_r \odot \overline{\mathbf{e}}_t\right)$	Pointwise Logistic
	SimplE	$\frac{1}{2}\left(\langle \mathbf{e}_h^h, \mathbf{e}_r, \mathbf{e}_t^t \rangle + \langle \mathbf{e}_h^t, \mathbf{e}_r^{-1}, \mathbf{e}_t^h \rangle\right)$	Pointwise Logistic
Convolutional	ConvE	$g\left(\mathrm{vec}\left(g\left(\mathrm{concat}\left(\widehat{\mathbf{e}}_h, \widehat{\mathbf{e}}_r\right) * \omega\right)\right)\mathbf{W}\right) \cdot \mathbf{e}_t$	Binary Cross-Entropy
	ConvKB	$\mathrm{concat}\left(g\left(\left[\mathbf{e}_h, \mathbf{e}_r, \mathbf{e}_t\right] * \omega\right)\right) \cdot \mathbf{w}$	Pointwise Logistic
	R-GCN	DistMult decoder	Binary Cross-Entropy
	CompGCN	ConvE decoder	Binary Cross-Entropy

eters provided by the original authors, when available. For the models with no reported best hyperparameters, as well as for the remaining datasets used in the experiments – *i.e.* DB93k, YAGO3-37k and YAGO4-19k – different combinations of hyperparameters were manually tried. We first trained our KGEMs for $1,000$ epochs, then noticed the best achieved results were found around epoch 400 or below. Consequently, we stick to a maximum of 400 epochs of training as in LibKGE (except R-GCN which is trained during $4,000$ epochs due to lower convergence to the best achieved results). For each positive triple in the training set, one corresponding negative triple is generated. To ensure fair comparisons between our models, we stick with UNS [18]. The chosen hyperparameters leading to the best performance on the validation dataset are provided as supplementary material (see Table B.2). Recall that the objective of this work is to perform a fair and insightful assessment of the semantic awareness of KGEMs. As such, the intended purpose was not to reach optimal performance in terms of rank-based metrics. Instead, the objective is to identify a set of hyperparameters that provides satisfying and stable performance, and then study how Sem@K behaves, both for a fixed epoch – *e.g.* for the best epoch in terms of MRR – and dynamically w.r.t. the number of epochs.

4.3.2 Results

In the following, we perform an extensive analysis of the results obtained using the aforedescribed KGEMs and KGs. For the sake of clarity, the complete range of tables and plots are placed in Appendix B.2 and B.3, respectively. When necessary to support our claim, some of them are duplicated in the content of the present section. Results achieved with the best reported hyperparameters are presented in Tables B.3 and B.4.

Semantic awareness of KGEMs

This section draws on the Sem@K values (see Tables B.3 and B.4) achieved at the best epoch in terms of MRR to provide an analysis of the semantic awareness of state-of-the-art KGEMs. In other words, for such models we only consider a snapshot of their best epochs in terms of rank-based metrics.

A major finding is that models performing well with respect to rank-based metrics are not necessarily the most competitive when it comes to their semantic capabilities. On YAGO3-37k (see Table 4.6) for instance, ConvE showcases impressive MRR and Hits@K values. However, it

Table 4.6: Evaluation of rank-based and semantic-oriented metrics on YAGO3-37k. Bold fonts and gray cells denote the best achieved results and the worst achieved results among the models reported in the table, respectively. Full results are available in Table B.3

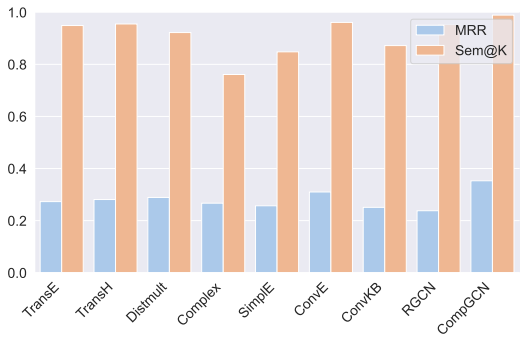
Model Family	Model	Rank-based				Sem@K(base)			Sem@K(wup)			Sem@K(ext)		
		MRR	H@1	H@3	H@10	S@1	S@3	S@10	S@1	S@3	S@10	S@1	S@3	S@10
Geometric	TransE	.184	.080	.198	.408	.989	.988	.988	.993	.994	.995	.897	.904	.911
	TransH	.187	.091	.199	.415	.995	.993	.993	.995	.997	.997	.901	.911	.925
Convolutional	ConvE	**.493**	**.350**	**.578**	**.775**	.933	.923	.910	.977	.977	.975	.893	.879	.871
	R-GCN	.115	.046	.110	.254	.993	.994	.993	.995	.996	.997	.933	.932	.928
	CompGCN	.399	.269	.464	.663	.998	.997	.996	.999	.999	.998	.998	.994	.982

is the worst KGEM in terms of Sem@K, as it is outperformed by CompGCN, R-GCN and all translational models, for all values of K.

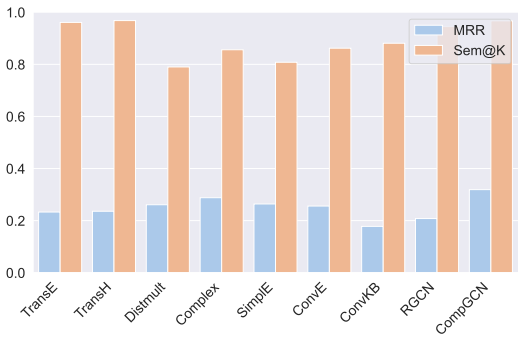
From a coarse-grained viewpoint, conclusions about the relative superiority of models with the distinct consideration of rank-based metrics and semantic awareness can be generalized at the level of models families. For example, semantic matching models (DistMult, ComplEx, SimplE) globally achieve better MRR and Hits@K values while their semantic capabilities are in most cases lower than the ones of translational models (TransE, TransH) – see Table B.3 and Table B.4 for detailed results w.r.t. rank-based and semantic-oriented metrics. A condensed view of the comparison between MRR and Sem@10 is also reported in Fig. 4.4 and Fig. 4.5. The respective ordering of such models for the benchmarked schema-defined and schemaless KGs are depicted in Fig. 4.6 and Fig. 4.7, respectively. It is clearly visible that KGEMs are grouped by family. In particular, GNNs and translational models showcase very promising semantic capabilities. GNNs are almost always the best regarding Sem@K(ext) values – not only for schemaless KGs (Fig. 4.7), but also for schema-defined KGs (see Table B.3 for full results, and Fig. 4.4 for a quick glimpse). This means GNNs are more capable of predicting entities that have been observed as head (resp. tail) of a given relation. Translational models are very competitive in terms of Sem@K(base). In Fig. 4.6, we clearly see that they consistently rank among the best performing models regarding Sem@K(base). Interestingly, the semantic matching models DistMult, ComplEx and SimplE perform relatively poorly. This observation holds regardless of the nature of the KG, as they systematically rank among the worst performing models for schema-defined (Fig. 4.6) and schemaless (Fig. 4.7) KGs.

Therefore, it seems that translational models are better able at recovering the semantics of entities and relations to properly predict entities that are in the domain (resp. range) of a given relation, while semantic matching models might sometimes be better at predicting entities already observed in the domain (resp. range) of a given relation (*e.g.* DistMult reaches very high Sem@K(ext) values on CoDEx-S, as evidenced in Fig. 4.7a). In cases when translational and semantic matching models provide similar results in terms of rank-based metrics, the nature of the dataset at hand – whether it is schema-defined or schemaless – might thus strongly influence the final choice of a KGEM.

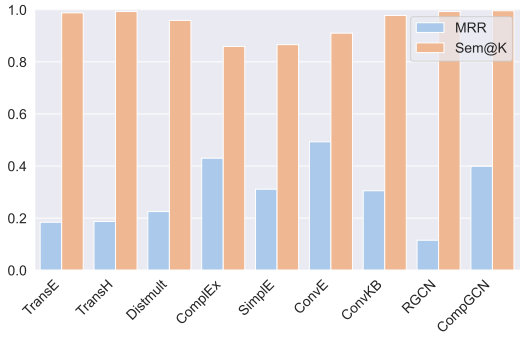
Interestingly, CompGCN which is by far the most recent and sophisticated model used in our experiments, outperforms all the other models in terms of semantic awareness as, with very limited exceptions, it is the best in terms of Sem@K(base), Sem@K(ext) and Sem@K(wup). In addition, it should be noted that R-GCN provides satisfying results as well. Except in a very few cases (*e.g.* outperformed by ComplEx on CoDEx-S and WN18RR in terms of Sem@1), R-GCN showcases better semantic awareness than semantic matching models. Most of the time,



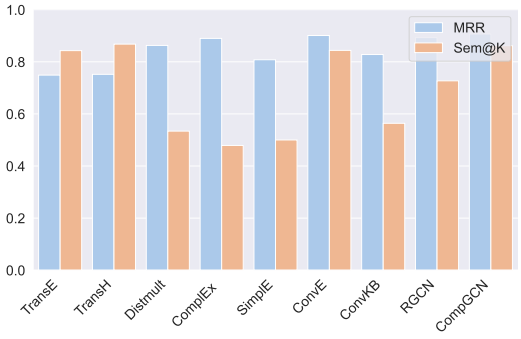
(a) FB15k237-ET



(b) DB93k

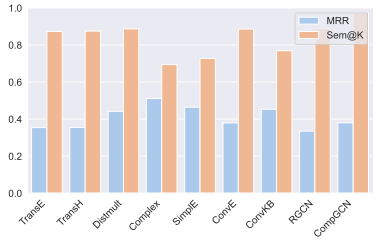


(c) YAGO3-37k

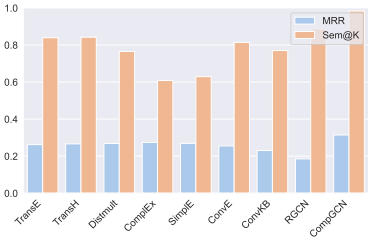


(d) YAGO4-19k

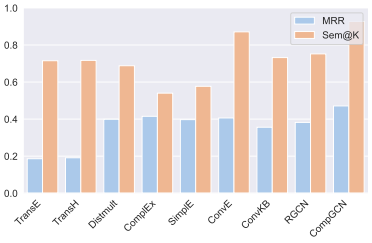
Figure 4.4: MRR and Sem@10 results achieved at the best epoch in terms of MRR for each model and on each schema-defined dataset



(a) CoDEx-S

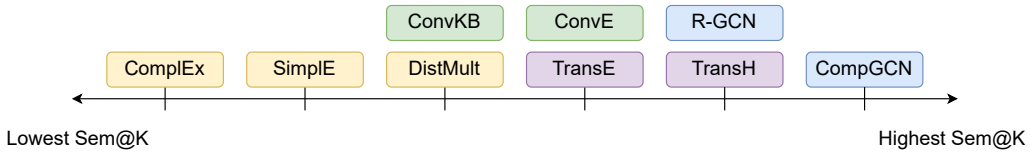


(b) CoDEx-M

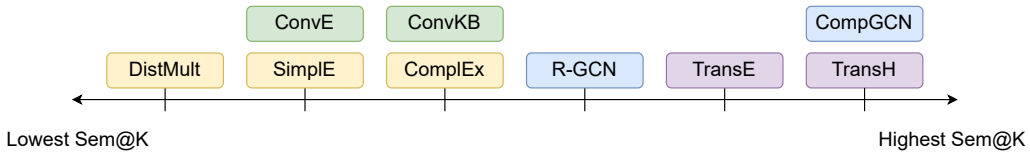


(c) WN18RR

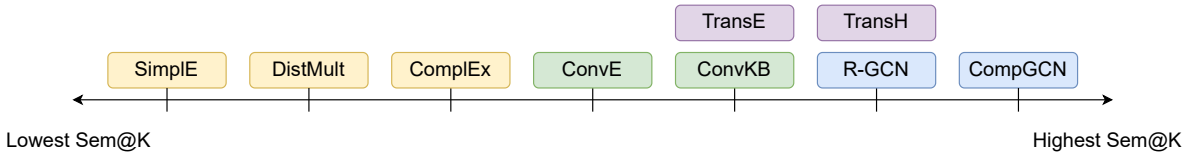
Figure 4.5: MRR and Sem@10 results achieved at the best epoch in terms of MRR for each model and on each schemaless dataset



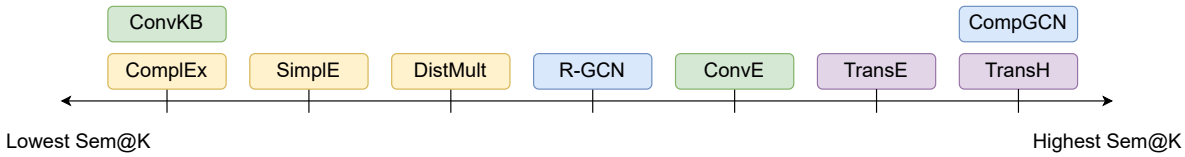
(a) FB15k237-ET



(b) DB93k

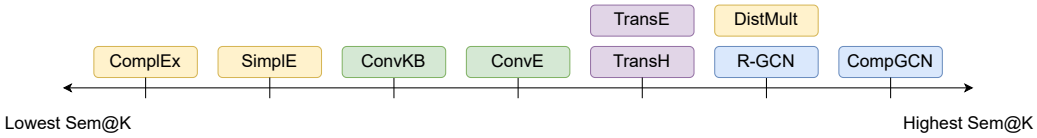


(c) YAGO3-37k

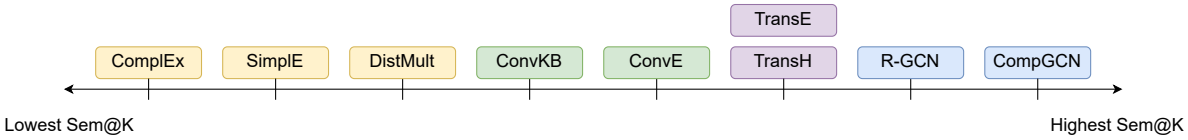


(d) YAGO4-19k

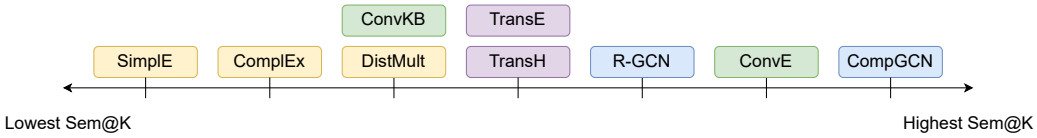
Figure 4.6: Sem@K(base) comparisons between KGEMs on the 4 benchmarked schema-defined KGs. Colors indicate the family of models: blue, purple, green, and yellow cells denote GNNs, translational, convolutional, and semantic matching models, respectively. Regarding Sem@K, the relative hierarchy of models is consistent across KGs and we can clearly see that KGEMs are grouped by families of models.



(a) CoDEx-S

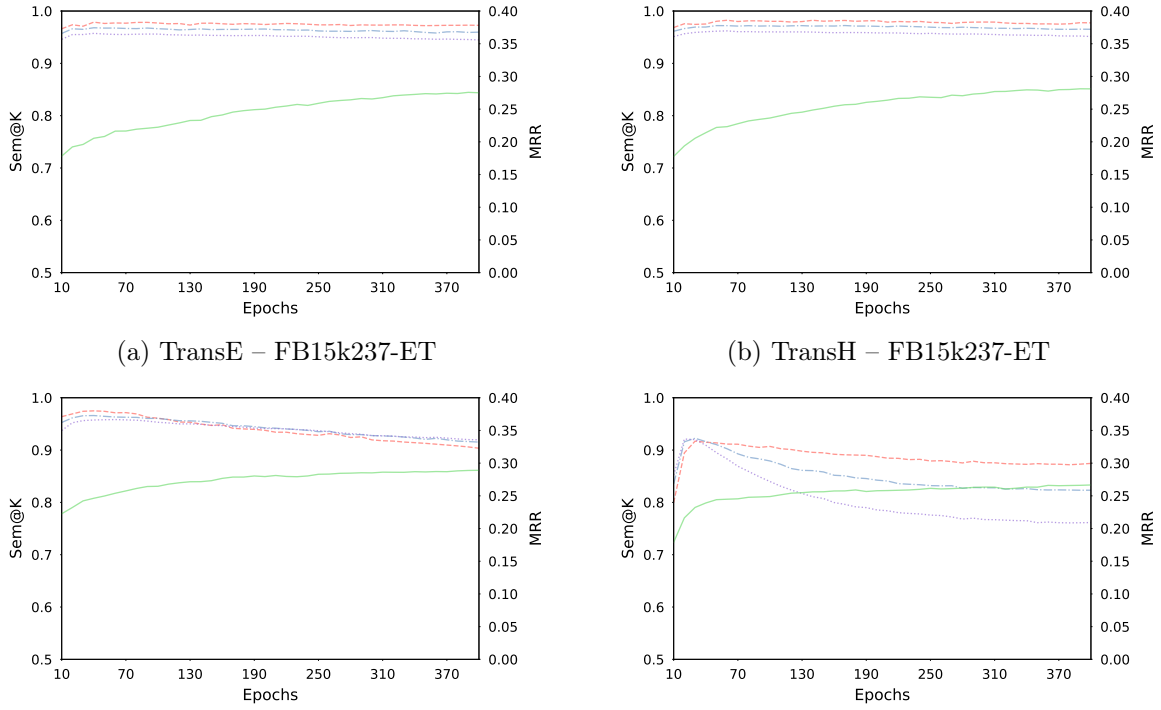


(b) CoDEx-M



(c) WN18RR

Figure 4.7: Sem@K(ext) comparisons between KGEMs on the 3 benchmarked schemaless KGs. Colors indicate the family of models: blue, purple, green, and yellow cells denote GNNs, translational, convolutional, and semantic matching models, respectively. Regarding Sem@K, the relative hierarchy of models is consistent across KGs and we can clearly see that KGEMs are grouped by families of models



(a) TransE – FB15k237-ET

(b) TransH – FB15k237-ET

(c) DistMult – FB15k237-ET

(d) ComplEx – FB15k237-ET

Figure 4.8: Evolution of MRR (—), Sem@1 (- -), Sem@3 (-·-), and Sem@10 (····) for translational (TransE, TransH) and semantic matching models (DistMult, ComplEx) on FB15k237-ET

R-GCN also provides comparable or even higher semantic capabilities than translational models. In particular, Sem@K values achieved with R-GCN are similar to the ones achieved with TransE and TransH (*e.g.* on FB15k237-ET and YAGO3-37k, see Tables B.3a and B.3c) while the latter models are actually outperformed by R-GCN in terms of Sem@K(ext). It appears clearly on CoDEx-M and WN18RR (Tables B.4b and B.4c), although the conclusion holds for all datasets.

Our experimental results suggest that the structure of GNNs seems to be able to encode the latent semantics of the entities and relations of the graph. This ability may be attributed to the fact that, contrary to translational models which only model the local neighborhood of each triple, GNNs update entity embeddings (and relation embeddings in the case of CompGCN) based on the information found in the extended, h-hop neighborhood of the central node. While translational and semantic matching models treat each triple independently, GNNs model interactions between entities on a large range of semantic relations. It is likely that this extended neighborhood comprises signals or patterns that help the model infer the classes of entities, thus providing very promising semantic capabilities in all experimental conditions.

Dynamic appraisal of KGEM semantic awareness

For certain models, rank-based metrics performance and semantic capabilities improve jointly. For others, the enhancement of their performance in terms of rank-based metrics comes at the expense of their semantic awareness. Interestingly, trends emerge relatively to families of models. First, we observe that a trade-off exists for semantic matching models. Results are particularly striking on FB15k237-ET (see Fig. 4.8), where it is obvious that after reaching the best Sem@K values after a few epochs, Sem@K values of DistMult and ComplEx quickly drop while

MRR continues rising. Conversely, translational models are more robust to Sem@K degradation throughout the epochs. Even though the best achieved Sem@K values are also reported in the very first epochs, once these values are reached they remain stable for the remaining epochs of training. This might be due to the geometric nature of such KGEMs, which will organize the representation space so as to $h + r$ falls in a region of the space where neighboring entities of the ground-truth tail t all are entities of the expected type. This is highly related to the block structure property, which is a common statistical pattern found in KGs [123]. It refers to the fact that entities can naturally belong to different groups (blocks), such that all the entities of a given group are linked to entities of another group through the same relationship. In this case, each group comprises entities of the same class. Translational models will naturally group entities of the same class in the same region of the representation space, as this is determined by the translation vector in that space.

Plots of the joint evolution of MRR and Sem@K values show that most of the KGEMs reaches their best Sem@K values after a few number of epochs. This means that predictions get semantically valid in the early stages of training. As previously mentioned, Sem@K then usually start to decrease, as it has been noted for semantic matching models in particular. To this respect, an excerpt of the head and tail predictions of DistMult on YAGO3-37k is depicted in Fig. 4.9. Even though the ground-truth entity does not show up in neither the head nor the tail top-K list, we clearly see that after only 30 epochs of training, predictions made by DistMult are more meaningful than after 400 epochs of training. This relative trade-off between making semantically valid predictions and predictions that comprise the ground-truth entity higher in the top-K list calls for finding a compromise in terms of training. The LP task is usually addressed in terms of rank-based metrics only, hence the choice of performing more and more training epochs so as to find the optimal KGEM in terms of MRR and Hits@K. However, as discussed in the present work, adding training steps may improve KGEM performance at the expense of its semantic awareness. In many cases, rank-based metrics values only slightly increase, whereas Sem@K values drastically drop. For instance, comparing MRR vs. Sem@K evolution of DistMult on FB15k237-ET (Fig. B.1c), we clearly see that after a moderate number of epochs, any additional epoch of training only provides a very slight improvement in terms of MRR, while it is very detrimental to Sem@K values. Depending on the use case, such a decline in the semantic capabilities of the model is not desirable, and a compromise is to be found between training more to increase KGEM predictive performance and stopping training early enough so as not to deteriorate its semantic awareness too much.

On the use of Sem@K for different kinds of KGs

As reported in Table 4.2, KGs based upon a schema and a class hierarchy are candidates for the computation of all the versions of Sem@K. For the schema-defined KGs used in our experiments, we choose to report values regarding all these metrics so as to enable multi-view comparisons across models. From Table B.3 it can be clearly seen that the relative superiority of models is consistent throughout the different Sem@K definitions. From a higher perspective, this means that even for schema-defined KGs with a hierarchy class, Sem@K(ext) is already a good proxy. This may be a good option to only rely on the Sem@K(ext) whenever the computation of Sem@K(base) is too expensive, due to the entity type checking part. This is even more true for Sem@K(wup), which requires an additional step of semantic relatedness computation.

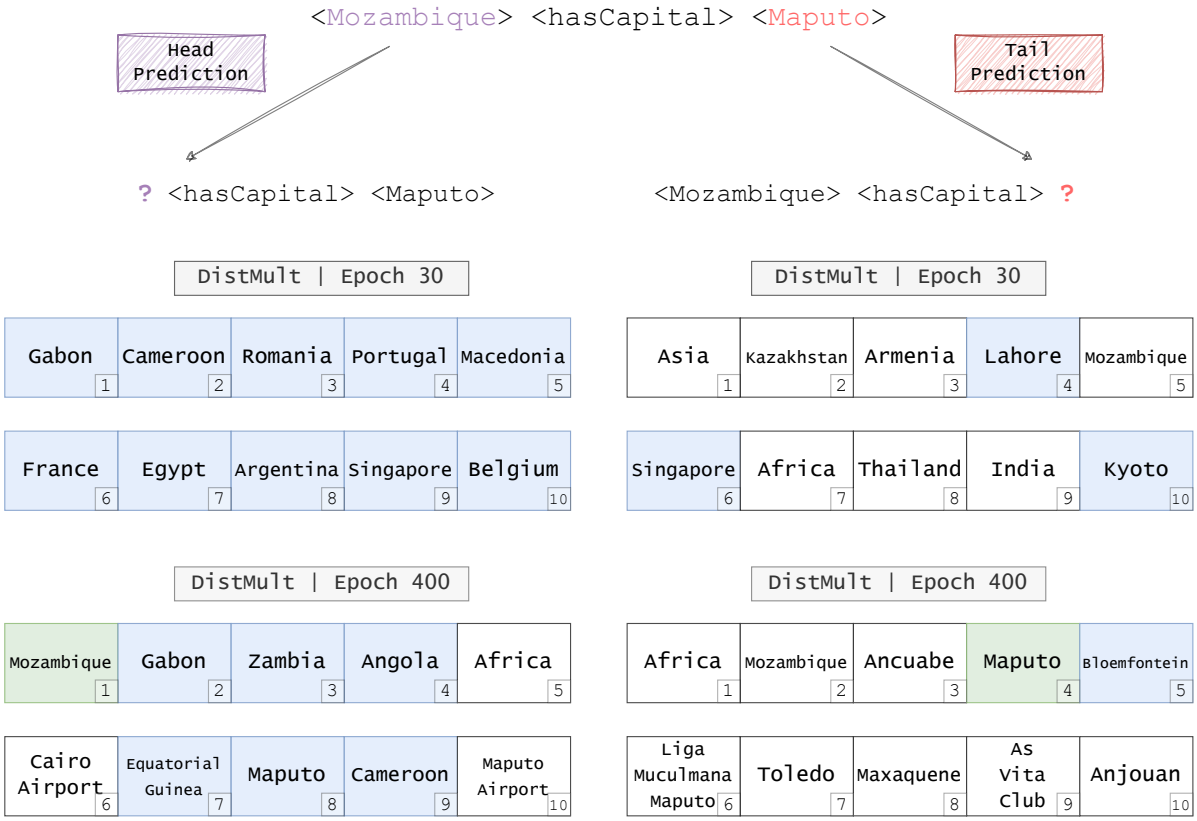


Figure 4.9: Top-ten ranked entities for head and tail predictions at epochs 30 and 400 for a sample triple from YAGO3-37k. Green, blue and white cells respectively denote the ground-truth entity, entities other than the ground-truth and semantically valid, and entities other than the ground-truth and semantically invalid. In this case, semantic validity is based on the domain and range of the relation `<hasCapital>`

4.3.3 Discussion

Three major research questions have been formulated in Section 4.1. Based on the analysis presented in Section 4.3.2, we discuss each research question individually – except **RQ2.4** which is specifically addressed in Section 4.4. We ultimately discuss the potential for further considerations of semantics into KGEMs.

RQ2.1. How the evaluation of KGEM semantic awareness should adapt to the typology of KGs?

Drawing on the initial version of Sem@K as presented in [75] – referred to herein as Sem@K(base) – an issue is quickly encountered when it comes to schemaless KGs, which do not contain any `rdfs:domain` (resp. `rdfs:range`) clause to indicate the class that candidate heads (resp. tails) should belong to. Our work introduces Sem@K(ext) – a new version of Sem@K that overcomes the aforementioned limitation. In addition, even with schema-defined KGs, Sem@K(base) is not necessarily sufficient in itself. This metric can be further enriched whenever a KG comes with a class hierarchy. In Section 4.2.5, we integrate class hierarchy into Sem@K by means of a similarity measure between concepts. We subsequently provide an example using the Wu-Palmer similarity score. The resulting Sem@K(wup) is used in the experiments in Section 4.3.2 and provide a finer-grained measure of KGEMs semantic awareness.

RQ2.2. How semantic-aware agnostic KGEMs are?

From a coarse-grained viewpoint, we noted that KGEMs trained in an agnostic way prove capable of giving higher scores to semantically valid triples. However, disparities exist between models. Interestingly, these disparities seem to derive from the family of such models. Globally, translational models and GNNs – represented by R-GCN and CompGCN in this work – provide promising results. It appears that the two aforementioned families of KGEMs are better able than semantic matching models (DistMult, ComplEx, SimplE) at recovering the semantics of entities and relations to give higher score to semantically valid triples. In fact, semantic matching models are almost systematically the worst performing models in terms of semantic awareness. From a dynamic standpoint, it is worth noting the high semantic capabilities of KGEMs reached during the first epochs of training. In most cases, this is even during the first epochs that the optimal semantic awareness is attained.

RQ2.3. Does the evaluation of KGEM semantic awareness offer different conclusions on the relative superiority of some KGEMs compared to rank-based metrics?

A major finding is that models performing well with respect to rank-based metrics are not necessarily the most competitive regarding their semantic capabilities. We previously noted that translational models globally showcase better Sem@K values compared to semantic matching models. Considering MRR and Hits@K, the opposite conclusion is often drawn. Hence, the performance of KGEMs in terms of rank-based metrics is not indicative of their semantic capabilities. The only exception that might exist is for GNNs that perform well both in terms of rank-based metrics and semantic-oriented measures.

The answers provided to the research questions also lead to consider new matters. As evidenced in Table B.3 and Table B.4, some KGs are more challenging with regard to Sem@K results. Due to its tailored extraction strategy that purposedly favored difficult relations to feature in the validation and test sets, YAGO4-19k is the schema-defined KG with the lowest

achieved Sem@K. This observation raises a deeper question: what characteristics of a KG make it inherently challenging for KGEMs to recover the semantics of entities and relations? An extensive study of the influence of KGs characteristics on the semantic capabilities of KGEMs would require to benchmark them on a broad set of KGs with varying dimensions, so as to determine those that are the most prevalent. Such characteristics can be the total number of relations, the average number of instances per class, or a combination of different factors. We leave this experimental study for future work.

Recall that this work is motivated by the possibility of going beyond a mere assessment of KGEM performance regarding rank-based metrics. We showed that these metrics only evaluate one aspect of such models, somehow providing a partial view on the quality of KGEMs. Our proposal for further assessing KGEM semantic capabilities aims at diving deeper into their predictive expressiveness and measuring to what extent their predictions are semantically valid. However, this second evaluation component does not shed full light on the respective KGEM peculiarities. Other evaluation components may be added, such as the storage and computational requirements of KGEMs [169, 137] and the environmental impact of their training and testing procedure [132]. Furthermore, the explainability of KGEMs is another dimension that deserves great attention [185, 143].

Towards further considerations of semantics in knowledge graph embeddings models

The Sem@K metric presented in this work allows for a more comprehensive evaluation of KGEMs. Based on domains and ranges of relations, Sem@K assesses to what extent the predictions of a model are semantically valid. The present work constitutes one of several stepping stones toward the further consideration of ontological and semantic information in KGEM design and evaluation.

It should be noted that due to the only consideration of domains and ranges, Sem@K cannot indicate whether predictions are logically consistent with other constraints posed by the ontology. This is in contrast with Inc@K presented in [87] that takes a broader set of ontological axioms into account. However, Inc@K and Sem@K intrinsically assess distinct dimensions of predictions. While the former is concerned with the logical consistency of predictions, the latter focuses on whether these predictions are semantically valid. For instance, an ontology can specify that `City` is the range of `livesIn`, that `Seattle` is a `City` but not a `Capital`, and that entities of type `President` should be linked to a `Capital` through the relation `livesIn`. Hence, it would still be meaningful and semantically correct to predict that `(BarackObama,livesIn,Seattle)`. However, this prediction is not logically consistent w.r.t. ontology specifications. Inc@K and Sem$K$ thus consider triples at different levels: while a given triple can be meaningful and semantically correct on its own, its combination with other triples may not be semantically valid or consistent with the ontology. In future work, we will consider more expressive ontologies and see how the broad collection of axioms that constitute them can be incorporated into Sem@K.

KGEMs evaluated in this work are all agnostic to ontological information in their design. However, some models that consider or ensure specific ontological or logical properties exist. For example, HAKE [188] is constructed with the purpose of preserving hierarchies, Logic Tensor Networks [6] are designed to ensure logical reasoning, and the training of TransOWL and TransROWL [35] is enriched with additional triples deduced from, *e.g.*, inverse predicates or equivalent classes. Because of the integration of semantic information in their design or training, one could wonder if they present improved semantic awareness compared to agnostic models. Additionally, KGEMs can also be used to predict triples that represent class instantiations. A possible extension of the present work thus consists in studying whether predicted links and class

instantiations are consistent and lead to increased Sem@K values. This would further qualify and highlight the semantic awareness and the consistency of predictions of KGEMs. We leave these questions for future work.

4.4 Application to recommender systems

In the precedent section, we discussed how Sem@K could be successfully used to track the semantic awareness of KGEMs. However, it still remains unclear how this semantic awareness would impact real-world applications.

In the present section, we aim at answering this. More concretely, we focus on recommender systems, and give an insight into how Sem@K can be used to assess how coherent or sensical the recommended items are.

Consequently, this whole section provides an answer for our final research question:

RQ2.4. How can Sem@K be used in real-world downstream tasks, *e.g.* recommendation?

It should be noted that many KG-based recommender systems are deployed in environments made of entities of different kinds. In the framework of the project AILES, we deal with KGs formed by high school and university students, curricula, cities, schools, assignments, topics, keywords of interest, etc. When it comes to recommending curricula to a given student, it makes sense that the system should not recommend anything else than curricula in the displayed list of recommendations. Going a step further, if most curricula are typed in a finer-grained way, it becomes possible to learn the profile and aspirations of a given student (*e.g.* by machine learning or other approaches) in order to recommend curricula of a particular kind (*e.g.*, only scientific-related curricula). Similarly, as we now have access not only to the label of each curriculum, but also to its area of specialization, it becomes possible to purposedly diversify the list of recommendations.

If we stick to those items which we know are properly labeled, we can create a dataset and train a KGEM on its train set. Computing Sem@K on triples in the test set, we then have a measure of how well the KGEM is able to perform this filtering step itself. Ultimately, a trained KGEM with good performance w.r.t. Sem@K can be deployed in production environment where new items do not necessarily have a proper category label. While basic filtering on this category label cannot be done, there is much chance that using the trained KGEM can retrieve adequate items to be further scored and ranked.

However, training a KGEM for the recommendation task needs further adaptations. In particular, one might want to approach the recommendation task as a LP task on a single target relation in the KG, which represents the link between users (*e.g.* students) and items (*e.g.* curricula) to recommend [43, 126]. Recommendation consists in predicting one or a few relevant items of interest for a current user. In other words, for a given triple $(u, r, ?)$, the goal is to recommend the more relevant items for user u, with r denoting the nature of the recommendation based on the application context. Note that there is only the need to perform tail predictions.

In [76], we instantiate this with two concrete use cases: recommending university curricula to high school students and recommending academic venues for researchers to publish their scientific articles. The datasets used to support the aforementioned use cases are EduKG and KG20C, respectively. EduKG is described in greater details in Section 3.3 as this resource was developed in the context of the project AILES, whereas Section 3.2.2 briefly describes the content of KG20C. Both datasets naturally lend themselves to the recommendation task, as they both feature one target relation of interest in their respective domain of application.

Table 4.7: Evaluation results on EduKG. Green cells indicate which strategy performs best for a given model and metric

	B-UNS			S-UNS		
	Sem@1	Sem@5	Sem@10	Sem@1	Sem@5	Sem@10
TransE	97.39	94.79	92.80	99.69	98.94	98.36
TransH	99.13	98.35	97.39	99.61	99.39	98.98
DisMult	95.77	95.23	95.02	97.17	96.75	96.45
ComplEx	82.31	79.47	77.80	98.69	97.94	97.61

Table 4.8: Evaluation results on KG20C. Green cells indicate which strategy performs best for a given model and metric

	B-UNS			S-UNS		
	Sem@1	Sem@5	Sem@10	Sem@1	Sem@5	Sem@10
TransE	43.90	27.15	20.59	93.68	83.59	72.62
TransH	97.42	86.46	69.87	99.82	99.26	93.54
DisMult	51.46	30.30	20.72	73.48	49.20	34.74
ComplEx	54.91	31.16	20.48	86.72	64.04	44.06

More concretely, for each triple of the form $(h, r, ?)$, in EduKG we aim at retrieving the ground-truth tail t where h is an entity of type `Student`, r accounts for the relation `likedCurriculum` and t is an entity of type `Curriculum`. An example would consist in retrieving the correct tail for the following test triple: (`Bob`, `likedCurriculum`, `?`).

Regarding KG20C, for each triple of the form $(h, r, ?)$, we aim at retrieving the ground-truth tail t where h is an entity of type `Paper`, r accounts for the relation `publishedIn` and t is an entity of type `Conference`. An example would be retrieving the correct tail for the following test triple: (`LearningToEfficientlyRank`, `publishedIn`, `?`). In this case, the ground-truth tail is `SIGIR`.

For each use case and associated dataset, Tables 4.7 and 4.8 report Sem@K values achieved with four distinct KGEMs under two different sampling strategies: (vanilla) UNS [19] on all relations – as it is done for the LP task –, and (specialized) UNS on the relation to recommend. The latter strategy implies that after performing (vanilla) UNS until best epoch or the triggering of early-stopping, the embedding model is refined by performing additional training epochs and applying UNS when corrupting the tails of triples in the filtered train set (*i.e.* triples featuring the target relation). These two strategies are named B-UNS and S-UNS, respectively (Tables 4.7 and 4.8).

For the two aforementioned use cases, the two following findings deserve highlighting:

1. **Analysis of B-UNS**. On EduKG, TransH stands out for its ability to assign higher scores to more relevant items where, in this case, relevancy is understood as the semantic validity of the recommended items w.r.t. the range of the target relation. For instance, on EduKG, relevant or semantically valid items are curricula. On KG20C, results are very low. Overall, these results suggest that with some models and on some datasets, KGEMs can act as decent filters.

2. **Analysis of S-UNS**. On EduKG and KG20C, we observe a systematic improvement of Sem@K values after resuming NS on the target relation only[24]. This result indicates that

[24]Note that we ensured optimal training of KGEMs under B-UNS so that the better results achieved with

KGEMs can act as even more effective filters if they are trained the right way.

More importantly, the latter result suggests that going beyond the traditional learning approaches can improve the semantic awareness of KGEMs. This result prompted us to investigate alternative training approaches, and motivates the following Chapter 5, where we discuss how schema-based information are not solely used for evaluation purposes, but can also be leveraged during training.

4.5 Conclusion and future work

4.5.1 Recap

In this chapter, we started by raising concerns about the sole reliance on rank-based metrics to evaluate link prediction models: not only using these metrics do not always help identify a single best-performing model, but these metrics are also intrinsically flawed. Inevitably, complementary metrics are needed to balance these limits and provide a more comprehensive evaluation framework for knowledge graph embedding models.

Focusing on the semantic awareness of such models, our first contribution was to introduce Sem@K, a novel semantic-oriented metric that can be instantiated with many variants based on the nature of the knowledge graph considered, and that measures to what extent a model is able to assign higher plausibility scores to candidates that comply with the domain (resp. range) of a relation.

After defining Sem@K, a subsequent and natural direction was to assess the semantic awareness of popular knowledge graph embedding models. Our second contribution lies in carrying out this effort. In particular, based on 7 datasets and 9 embedding-based models, we conducted an extensive re-evaluation of these models with regards to Sem@K. Our provided analysis brought many insights into how distinct embedding-based models behaved in terms of semantic awareness. Notably, we have shown that the performance of models in terms of rank-based metrics does not pre-empt their performance in terms of Sem@K. Therefore, Sem@K appears to be a complementary metric to the traditionally used rank-based metrics and brings an extra dimension to the current evaluation framework of knowledge graph embeddings.

Finally, our last contribution was to provide a real use case of Sem@K, highlighting its need in some particular fields such as item recommendations. Drawing on two concrete examples – recommending curricula to students and recommending venues for researchers to publish their articles – we pointed out the necessity of predicting items of a certain type, *i.e.* items that comply with the range of the recommendation. Importantly, Sem$K$ can act as a filter in some situations, thereby reducing the need for a built-in filtering procedure that is inherently limited by data quality issues.

4.5.2 Future work

This chapter laid out a foundational basis on how measuring the semantic capabilities of KGEMs. As such, there are still many open issues and challenges to consider for future work. We discuss some of them below:

> **Experimental coverage.** Our re-evaluation of KGEMs in terms of Sem@K on numerous datasets – although extensive – is not systematic. Consequently, in future works we will extend our analysis using even more datasets and models.

S-UNS are not simply due to performing more training epochs.

Handling incomplete information. Recall that the datasets we used were carefully filtered so that all relations have properly defined domains and ranges. Although this was desirable for experimental purposes, it may not correspond to real-world use cases with evolving domain and range information as well as relations with incomplete (*i.e.* only a domain, or a range) or missing profile altogether. In future work, we will try to adapt our approach to information heterogeneity.

Extending Sem@K. From a general viewpoint, Sem@K focuses on domains and ranges to evaluate KGEM semantic awareness. This can be seen as a limitation for fully assessing the semantic capabilities of KGEMs, which are not limited to predicting entities complying with relations' profiles. Consequently, we will design more comprehensive versions of Sem@K that encompass a wider set of ontological properties.

Chapter 5

Enhancing knowledge graph embeddings with schema-based information

Contents

This chapter covers the following articles:

Schema First! Learn Versatile Knowledge Graph Embeddings by Capturing Semantics with MASCHInE. Hubert *et al.* (2023). Proceedings of the 12th Knowledge Capture Conference 2023, K-CAP 2023, Pensacola, FL, USA, December 5-7, 2023.

Treat Different Negatives Differently: Enriching Loss Functions with Domain and Range Constraints for Link Prediction. Hubert *et al.* (2024). The Semantic Web - 21st International Conference, ESWC 2024, Hersonissos, Crete, Greece, May 26 - May 30, 2024. (**Best Student Research Paper Award**)

New Strategies for Learning Knowledge Graph Embeddings: the Recommendation Case. Hubert *et al.* (2022). Knowledge Engineering and Knowledge Management - 23rd International Conference, EKAW 2022, Bolzano, Italy, September 26-29, 2022. (**Best Paper Award**)

5.1 Motivations

Chapter 4 addressed the need for a multi-faceted evaluation framework for KGEMs. In particular, our first contribution was to introduce Sem@K – a novel semantic-oriented metric to assess the semantic awareness of such models. Our experimental results highlighted the importance of measuring this aspect of KGEMs – along with traditional rank-based metrics – in order to have a fuller picture of KGEM performance and be able to differentiate between KGEMs with similar performance in terms of rank-based metrics. It is worth pointing out that measuring Sem@K and more specifically Sem@K(base) is made possible by the presence of schema-based information about relation domains and ranges, which further strengthens the need for semantically rich datasets, as discussed in Chapter 3.

In parallel to using information about relation domains and ranges from the schema to assess models' semantic awareness, several approaches leverage schema-based information to improve model performance. Over the past few years, semantic information has been incorporated in the training procedure of KGEMs [97, 35, 87]. Even though such neuro-symbolic approaches are commendable, they use schema-based information with the primary goal of improving KGEM performance w.r.t. rank-based metrics. It would be desirable to also measure their semantic awareness, *e.g.* using Sem@K. Intuitively, one may think that injecting information from the schema at training time would inevitably improve Sem@K values. However, this general intuition should not be taken for granted. First, there is no guarantee that this is effectively the case, and lower Sem@K values achieved while incorporating schema-related information would be an interesting result to discuss, as it would imply that such approach does not necessarily lead to better semantic awareness and that further investigation is needed. Second, it remains to assess whether an increased semantic awareness comes at the expense of other dimensions of KGEM evaluation (*e.g.* traditional rank-based metrics), and to what extent the impacts of injecting schema-based information differ between models, datasets, and training components. Actually, information extracted from the schema can be injected in different components of the full KGEM

pipeline (Fig. 2.1): in the interaction function, at initialization, in the negative sampler, or in the loss functions.

More precisely, this chapter studies the impact of schema injection in three components: in the negative sampling (Section 5.2), in the loss function (Section 5.3.1), and at embedding initialization (Section 5.4). These experiments all seek to answer the following research question:

> **RQ3.** On the basis of Sem@K and using semantically rich datasets, can we design neuro-symbolic training approaches to effectively leverage the wealth of available symbolic knowledge?

As in the precedent chapters, we address this intricate research question by designing smaller research questions:

> **RQ3.1.** How to use schema-based information during negative sampling to propose a recommender system based on KGEM for more accurate and semantically valid recommendations?
>
> **RQ3.2.** How main loss functions used in LP can incorporate domain and range constraints to differently consider negative triples based on their semantic validity and what is their impact on the overall KGEM performance?
>
> **RQ3.3.** How to use class-based information to efficiently initialize entity and relation embeddings and how profitable is it for KG-based tasks?

The main contributions of this chapter are:

1. We propose to use type-constrained negative sampling (TCNS) in the specific application context of recommender systems, and study the impact of TCNS on the accuracy and semantic validity in recommendations (Section 5.2) (**RQ3.1**).

2. We propose signature-driven versions for the three mostly used loss functions for the LP task – leveraging schema information about relation domains and ranges – and evaluate our approach both in terms of rank-based metrics and Sem@K. Notably, our results indicate that the designed signature-driven loss functions provide, in most cases, better performance w.r.t. both rank-based metrics and Sem@K and that signature information is worth incorporating into loss functions (Section 5.3.1) (**RQ3.2**).

3. We propose a conceptually simple yet efficient approach for initializing entity embeddings using class-based information extracted from a schema, and analyze the impact of fine-tuning and using the resulting embeddings in several KG-based tasks (Section 5.4) (**RQ3.3**).

5.2 Type-constrained negative sampling for enhanced recommendations

In Section 2.3.3, negative sampling has been extensively discussed and many approaches were presented. In particular, we started by presenting UNS [19] which samples random entities with uniform probability. We highlighted the undesirable likelihood of generating positive triples with UNS. For example, replacing the tail of (`CristianoRonaldo`, `playedFor`, `RealMadrid`)

with `ManchesterUnited` would generate the triple (`CristianoRonaldo`, `playedFor`, `ManchesterUnited`) which actually represents a true fact. Then, we presented BNS [174] as a more sophisticated NS based on Bernoulli sampling. Interestingly, this NS procedure reduces the risk of creating false-negative triples by setting different probabilities for replacing the head and the tail based on the nature of the relation that links them: if the relation r is 1-to-N (e.g. `parentOf`), the head h has a higher probability of being replaced. If the relation r is N-to-1 (e.g. `bornIn`), the tail t is more likely to be replaced.

However, the two aforementioned approaches cannot prevent the sampling procedure from producing semantically incorrect triples such as (`CristianoRonaldo`, `playedFor`, `JoeBiden`). Such nonsensical triples do not provide the model with sufficient signal to learn from, which causes the notorious zero loss problem. More elaborated methods presented in Section 2.3.3 such as NSCaching [186] and KBGAN [27] are proposed to address this zero loss issue. However, they are not immune to the problem of sampling false-negative triples.

Consequently, a few works incorporate semantic information to constrain the NS procedure and generate meaningful negative triples [97, 175, 87, 106]. For instance, type-constrained NS (TCNS) [97] replaces the head or the tail of a triple with a random entity belonging to the same type (`rdf:type`) as the ground-truth entity. By doing so, only semantically valid triples are generated during negative sampling. TCNS has been found to work better than pure RNS in several scenarios [96, 97]. Jain *et al.* [87] go a step further and use ontological reasoning to iteratively improve KGEM performance by retraining the model on inconsistent predictions. More recently, [106] propose hierarchical type enhanced NS (HTENS), which leverages hierarchical entity type information and entity-relation cooccurrence information to optimize the sampling probability distribution of negative samples.

To the best of our knowledge, all these works are concerned with the task of LP and have not been studied in the specific context of single-relation link prediction, especially recommendation. However, Grad-Gyenge *et al.* [57] compared traditional collaborative filtering and embedding models for making recommendations over KGs. They clearly showed that using KGEMs significantly improves recommendation performance without suffering from an increasing amount of user interactions, contrary to traditional collaborative filtering algorithms. Palumbo *et al.* [126] also demonstrated the superiority of using KGEMs over traditional baselines in a recommendation framework.

As a result, this section builds on Section 4.4 – where we showed a concrete application of Sem@K for RS-based applications – and positions itself at the intersection of the following two research fields: enhanced NS and recommendation with KGEMs. This section raises the question of whether performing TCNS enhances both recommendation accuracy and semantic validity, just as it does for the more general task of LP.

5.2.1 Training Embedding Models for Recommendation

As mentioned in Section 4.4, training a KGEM for the recommendation task needs some adaptations compared to the traditional LP task. In particular, we are interested in one single target relation. Consequently, the test set should only contain triples featuring this relation. In addition, different NS procedures can be considered, and in Section 4.4 we gave details about B-UNS and S-UNS.

Although S-UNS has some specificities compared to the traditional UNS [19] performed on all relations, it still remains schema-agnostic in the sense that all the observed entities in the KG are valid candidates in the corrupted triples.

In contrast, this chapter studies the injection of schema-based information while training

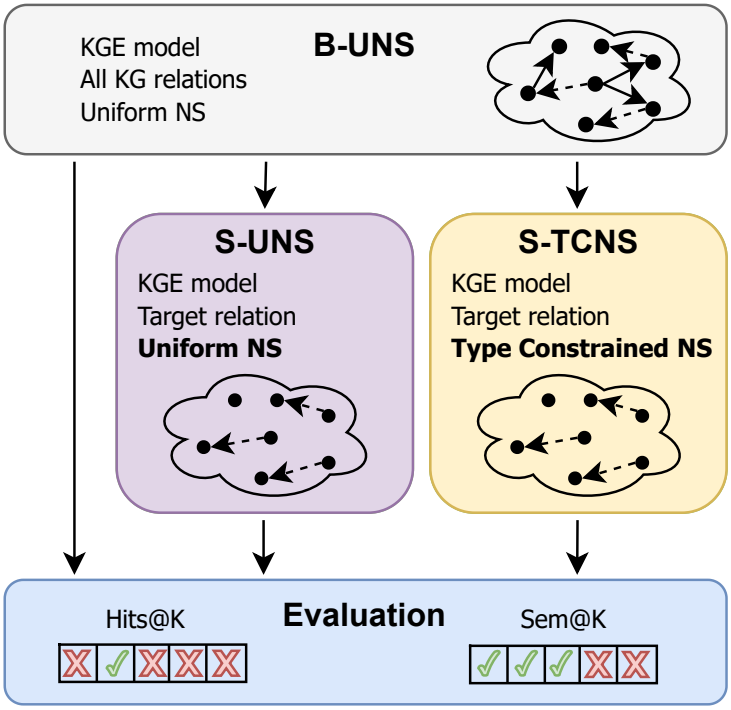


Figure 5.1: Proposed negative sampling strategies for the recommendation task

KGEMs, and the present section is more specifically concerned with doing so in the NS procedure. A common strategy is to leverage information about relations' ranges and the types of entities. This is inspired by the TCNS [97] which constraints corrupted head (resp. tail) entities to comply with the domain (resp. range) of a given relation. In the recommendation framework, we only corrupt tail entities for a given relation.

We introduce S-TCNS and define it as follows: S-TCNS refines a pre-trained embedding model on the target relation (as S-UNS does) by applying TCNS when corrupting the tails of the filtered triples in the train set, *i.e.* those triples that feature the target relation. In other words, we take the embeddings trained using B-UNS. Then, these embeddings are further trained for additional epochs by filtering out triples that do not contain the target relation, and resuming training on the remaining set of triples while enforcing a type-constrained negative sampling. Fig. 5.1 summarizes the different NS strategies introduced so far.

5.2.2 Results

Datasets. As in Section 4.4, we use EduKG and KG20C in the following experiments. We already mentioned that these datasets naturally lend themselves to the recommendation task. In addition, we should highlight their adequate entity typing: given a head-relation pair (h, r), the missing tail can only be of one single type. Consequently, we are able to study the influence of TCNS [97] on recommendation accuracy and semantic validity.

Regarding EduKG, recall that for each triple of the form $(h, r, ?)$, we aim at retrieving the ground-truth tail t where h is an entity of type `Student`, r accounts for the relation `likedCurriculum` and t is an entity of type `Curriculum`. Hence, for the S-TCNS strategy, only entities of type `Curriculum` are used to replace tails of positive triples during negative sampling.

Table 5.1: Evaluation results on EduKG. H@K and S@K stand for Hits@K and Sem@K respectively, for $K \in \{1, 5, 10\}$. Green cells indicate which strategy performs best for a given model and metric

	B-UNS			S-UNS			S-TCNS		
	H@1	H@5	H@10	H@1	H@5	H@10	H@1	H@5	H@10
TransE	4.66	16.12	25.49	6.27	22.66	33.64	8.19	26.75	37.82
TransH	7.19	22.96	35.51	9.67	30.02	43.49	12.16	34.38	47.80
DisMult	5.40	18.35	29.11	5.75	19.91	31.55	6.06	21.70	34.03
ComplEx	5.14	17.91	26.62	7.28	21.27	34.12	7.10	25.14	38.08
	S@1	S@5	S@10	S@1	S@5	S@10	S@1	S@5	S@10
TransE	97.39	94.79	92.80	99.69	98.94	98.36	96.21	91.62	88.19
TransH	99.13	98.35	97.39	99.61	99.39	98.98	99.17	97.90	96.17
DisMult	95.77	95.23	95.02	97.17	96.75	96.45	97.04	96.44	96.01
ComplEx	82.31	79.47	77.80	98.69	97.94	97.61	98.48	96.99	95.39

Regarding KG20C, the same rationale applies: we aim at retrieving the ground-truth tail t where h is an entity of type `Paper`, r accounts for the relation `publishedIn` and t is an entity of type `Conference`. Hence, for the S-TCNS strategy, only entities of type `Conference` are used to replace tails of positive triples during negative sampling.

Models and setup. Now that B-UNS, S-UNS, and S-TCNS are introduced, the full experimental procedure can be detailed in greater depth. In particular, the three NS strategies will be compared in terms of performance w.r.t. Hits@K and Sem@K metrics, for $K \in \{1, 5, 10\}$. Notably, Hits@K and Sem@K are only assessed on tail predictions. The experiments are conducted using a 5-fold cross-validation. In each cross-validation setting, the test fold only comprises triples whose relation r is the target relation. TransE, TransH, DistMult, and ComplEx are implemented using PyTorch. Importantly, when comparing different training strategies, models are instantiated with the same initialization seed. The choice of the number of epochs, negative samples, embedding dimensions, and learning rate are based on what was found to work best for these datasets. Regardless of the strategy used, we first perform 1,000 epochs of general training with early-stopping (B-UNS) to ensure that training is achieved in a reasonable amount of time and that the best evaluation metrics are recorded. 100 epochs of specialized training are subsequently performed with early-stopping by applying the two strategies S-UNS and S-TCNS. All models are trained using max-margin loss and Adam optimizer as in [96]. The hyperparameters are shared across all configurations to ensure a fair comparison of their respective performance: number of negative triples per positive one $C = 50$ (in accordance with [165]), embedding dimension $k = 20$, learning rate $\eta = 0.01$, margin $\gamma = 1.0$. For TransE and TransH, distance $d = L2$ was used.

Results are reported in Tables 5.1 and 5.2. Note that these tables share common information already presented in Tables 4.7 and 4.8. For the sake of clarity, in this section we report the full results. However, while Section 4.4 focused on the use of Sem@K for the recommendation task and only mentioned B-UNS and S-UNS, in this section we are concerned about how the inclusion of schema-based information about the target relation domain and entities' types in the NS procesure impacts recommendation performance. As such, in this section we instead consider how S-TCNS fares compared to B-UNS and S-UNS.

Table 5.2: Evaluation results on KG20C. H@K and S@K stand for Hits@K and Sem@K respectively, for $K \in \{1, 5, 10\}$. Green cells indicate which strategy performs best for a given model and metric

	B-UNS			S-UNS			S-TCNS		
	H@1	H@5	H@10	H@1	H@5	H@10	H@1	H@5	H@10
TransE	16.21	33.48	41.58	27.33	62.97	76.08	16.37	33.84	41.92
TransH	44.56	78.99	87.29	46.50	84.14	94.51	47.55	80.89	87.41
DisMult	23.05	36.57	40.89	28.54	46.20	51.97	27.67	43.03	47.37
ComplEx	24.98	39.54	44.40	32.21	57.28	63.03	27.45	49.16	58.87
	S@1	S@5	S@10	S@1	S@5	S@10	S@1	S@5	S@10
TransE	43.90	27.15	20.59	93.68	83.59	72.62	36.91	21.95	16.26
TransH	97.42	86.46	69.87	99.82	99.26	93.54	96.15	75.36	54.81
DisMult	51.46	30.30	20.72	73.48	49.20	34.74	67.77	38.59	24.22
ComplEx	54.91	31.16	20.48	86.72	64.04	44.06	74.53	59.84	47.36

5.2.3 Discussion

In this section, we dissect our experimental results to answer **RQ3.1.**, that we recall below:

> **RQ3.1.** How to use schema-based information during negative sampling to propose a recommender system based on KGEM for more accurate and semantically valid recommendations?

To answer **RQ3.1.**, we discuss Sem@K results on all models with the three strategies B-RNS, S-RNS and S-TCNS. In particular, we focus on comparing S-TCNS with the other two schema-agnostic strategies.

Focusing on the impact of S-TCNS on Sem@K, we note that on both datasets, the combined use of this strategy with the two translational models TransE and TransH deteriorates the model ability to capture the semantic profile of the target relation, not only compared to S-UNS, but also compared to B-UNS. On the contrary, using S-TCNS enhances Sem@K of DistMult and ComplEx.

Therefore, even though integrating the entity type into the negative sampling (S-TCNS) *can* improve the ability of a model to capture the semantic profile of the target relation, it is not always the case.

In Section 5.1, we presented the commonly shared intuition that injecting schema-based information at training time would inevitably improve Sem@K values, and we raised concerns about this assumption. Experimental results analyzed in this section clearly demonstrate this. As schema-based injections do not necessarily improve Sem@K, it further strengthens the need to investigate in which situations this is the case, and in which it is not. The following Sections 5.3 and 5.4 are underpinned by this motivation, and discuss the impacts of other schema-based approaches on KGEM performance.

5.3 Enriching loss functions with domain and range constraints for link prediction

In the precedent section, schema-based information was incorporated during negative sampling. As mentioned in Section 5.1, there are many other ways ontological knowledge can be considered. In particular, existing works proposing to include semantic information into loss functions

showcase promising results. In [60], entity embeddings of the same semantic category are enforced to lie in a close neighborhood of the embedding space. Likewise, d'Amato *et al.* [35] inject ontological knowledge in the form of `equivalentClass`, `equivalentProperty`, `inverseOf`, and `subClassOf` axioms, similarly to [111] who incorporate `equivalentProperty` and `inverseOf` axioms as regularization terms in the loss function. Cao *et al.* [29] propose a new regularizer called Equivariance Regularizer, which limits overfitting by using semantic information. However, they are restricted to specific loss functions [35, 60], or consider ontological axioms [35, 111] that do not include domain and range axioms (or relation signatures) which are widely available in KGs.

Interestingly, such axioms could be leveraged to generate different kinds of negative triples, *i.e.*, negative triples that respect relation signatures (called semantically valid) and those that do not (called semantically invalid).

Additionally, to the best of our knowledge, negative triples were only studied in the sampling procedure, where specific ones are chosen to train on and the others are discarded. The possibility to sample all kinds of negatives but differently consider them within loss functions was left unassessed.

Precedent work also pointed out the performance gain of incorporating ontological information as measured by rank-based metrics such as MRR and Hits@K [29, 35, 60, 111]. However, while such approaches include semantic information as KGEM inputs, the semantic capabilities of the resulting KGEM are left unassessed, even though this would provide a fuller picture of its performance [75, 79]. Hence, this twofold observation motivates **RQ3.2**:

> **RQ3.2.** How main loss functions used in LP can incorporate domain and range constraints to differently consider negative triples based on their semantic validity and what is the impact of on the overall KGEM performance?

To address this question, we propose signature-driven loss functions, *i.e.* loss functions containing terms that depend on some background knowledge about types of entities and domains and ranges of relations.

To broaden the impact of our approach, our work is concerned with the three most encountered loss functions in the literature: the pairwise hinge loss (PHL) [18], the 1-N binary cross-entropy loss (BCEL) [39], and the pointwise logistic loss (PLL) [164] (further detailed in Section 2.3.4). For each of them, we propose a tailored signature-driven version in Section 5.3.1. The considered BK is available in many schema-defined KGs [40], which makes these newly introduced loss functions widely usable in practice.

5.3.1 Signature-driven loss functions

In the present section, we consider the three most commonly used loss functions for performing LP [145]: the pairwise hinge loss (PHL) [18], the 1-N binary cross-entropy loss (BCEL) [39], and the pointwise logistic loss (PLL) [164]. Their vanilla formulas can be found in Section 2.3.4 (Equations (2.8), (2.9), and (2.10), respectively).

We propose signature-driven loss functions that extend the three most frequently used loss functions [145] and leverage BK about domains and ranges of relations, which are provided in many KGs used in the literature.

The goal of the proposed loss functions is to distinguish *semantically valid negatives* from *semantically invalid ones*. The former are defined as triples (h, r, t) respecting both the domain and range of the relation r, *i.e.*,

$$\text{type}(h) \cap \text{domain}(r) \neq \emptyset \wedge \text{type}(t) \cap \text{range}(r) \neq \emptyset$$

whereas the latter violate at least one of the constraints, *i.e.*

$$\text{type}(h) \cap \text{domain}(r) = \emptyset \vee \text{type}(t) \cap \text{range}(r) = \emptyset.$$

domain(r) (resp. range(r)) is defined as the expected type as head (resp. tail) for the relation r. For example, the relation `presidentOf` expects a `Person` as head and a `Country` as tail. Starting from a positive triple (`EmmanuelMacron`, `presidentOf`, `France`) which represents a true fact, (`BarackObama`, `presidentOf`, `France`) and (`EmmanuelMacron`, `presidentOf`, `Germany`) are examples of semantically valid negative triples, whereas (`Adidas`, `presidentOf`, `France`) and (`EmmanuelMacron`, `presidentOf`, `Christmas`) are examples of semantically invalid negative triples. In this work, entities are multi-typed. Therefore, type(e) returns the set of types (a.k.a. classes) that the entity e belongs to.

We introduce a loss-independent ϵ factor, which is dubbed as the *semantic factor* and aims at bringing the scores of semantically valid negative triples closer to the positive ones. This common semantic factor fitting into different loss functions shows the generality of our approach that can possibly be extended to other loss functions. Interestingly, this factor also allows to take into account to some extent the Open World Assumption (OWA) under which KGs are represented. Under the OWA, triples that are not represented in a KG are either false or missing positive triples. In traditional training procedures, these triples are indiscriminately considered negative, which corresponds to the Closed World Assumption. On the contrary, our proposal considers semantically invalid triples as true negative while semantically valid triples (and possibly missing positive or false negative under the OWA) are closer to true positive triples. This assumes that entity types are complete and correct.

$\mathcal{L}_{\mathbf{PHL}}$ defined in Equation (2.8) relies on the margin hyperparameter γ. Increasing (resp. descreasing) the value of γ will increase (resp. descrease) the margin that will be set between the scores of positive and negative triples. However, this unique γ treats all negative triples indifferently: the same margin will separate the scores of semantically valid and semantically invalid negative triples from the score of the positive triple they both originate from. We suggest that the scores of these two kinds of negative triples should be treated differently. Hence, our approach redefines $\mathcal{L}_{PHL}$ as follows (Equation (5.1)):

$$\mathcal{L}^S_{PHL} = \sum_{t \in \mathcal{T}^+} \sum_{t' \in \mathcal{T}^-} \left[\gamma \cdot \ell(t') + f(t') - f(t)\right]_+ \tag{5.1}$$

$$\text{where } \ell(t') = \begin{cases} 1 \text{ if } t' \text{ is semantically invalid} \\ \epsilon \text{ otherwise} \end{cases}$$

The loss function in Equation (5.1) has a superscripted S to make it clear this is the signature-driven version of the vanilla $\mathcal{L}_{PHL}$ as defined in Equation (2.8). A choice of $\epsilon < 1$ leads the KGEM to apply a higher margin between scores of positive and semantically invalid triples than between positive and semantically valid ones. For a given positive triple, this allows to keep the scores of its semantically valid negative counterparts relatively closer compared to the scores of its semantically invalid counterparts. Intuitively, when the KGEM outputs wrong predictions, more of them are still expected to meet the domain and range constraints imposed by relations. Thus, wrong predictions are assumed to be more meaningful, and, in a sense, semantically closer to the ground-truth triple.

$\mathcal{L}_{\mathbf{BCEL}}$ defined in Equation (2.9) is adapted to $\mathcal{L}^S_{BCEL}$ by redefining the labelling function ℓ. In particular, when dealing with a KG featuring typed entities and providing information about

domains and ranges of relations, the labelling function ℓ is no longer binary. Instead, the labels of semantically valid negative triples can be fixed to some intermediate value between the label value of positive triples and of semantically invalid negative triples, which leads to the labelling function ℓ defined in Equation (5.2):

$$\ell(t') = \begin{cases} 1 & \text{if } t' \in \mathcal{T}^+ \\ \epsilon & \text{if } t' \in \mathcal{T}^- \text{ and } t' \text{ is semantically valid} \\ 0 & \text{if } t' \in \mathcal{T}^- \text{ and } t' \text{ is semantically invalid} \end{cases} \tag{5.2}$$

where the semantic factor ϵ is a tunable hyperparameter denoting the label value of semantically valid negative triples. The intuition underlying the refinement of the labelling function ℓ is to voluntarily cause some confusion between semantically valid negative triples and positive triples. By bridging their respective label values, it is expected that the KGEM will somehow consider the former as "less negative triples" and assign them a higher score compared to positive triples.

$\mathcal{L}_{\mathbf{PLL}}$ defined in Equation (2.10) could be adapted to $\mathcal{L}^S_{PLL}$ similarly to $\mathcal{L}^S_{BCEL}$. In other words, the labelling function ℓ could also output an intermediate label value ϵ for semantically valid negative triples. Although this approach provides very good results in terms of Sem@K values, it does not provide consistently good results across datasets. Furthermore, obtained results in terms of MRR and Hits@K can be far below the ones obtained with the vanilla model (see supplementary materials for further details). That is why, here, to treat semantically valid and invalid negative triples differently, instead of modifying the labelling function ℓ, the semantic factor ϵ for $\mathcal{L}^S_{PLL}$ defines the probability with which semantically valid negative triples are considered as positive triples and therefore are labelled the same way. For example, with $\epsilon = 0.05$, at each training epoch and for each batch, the semantically valid negative triples of the given training batch will be considered positive with a probability of 5%.

It is noteworthy that our approach can be applied in practice to KGs with or without types, domains and ranges. Indeed, in the absence of such background knowledge, our signature-driven loss functions reduce to their respective vanilla counterparts. Recall that our approach does not focus on negative sampling or complex negative sample generators such as KBGAN [27], NSCaching [186], and self-adversarial negative sampling [157]. Although these works are related to ours, they constrain negative sampling upstream. On the contrary, our approach does not constrain the sampling of negative triples but rather dynamically distributes the negative triples into different parts of the loss functions, based on their semantic validity.

5.3.2 Experimental Setting

Datasets In these experiments, three datasets are considered: FB15k237, DB93k, and YAGO4-19k. While these datasets are described in greater depth in Section 3.2.2, Table 5.3 provide summary statistics about them. It is worth noting that in all of the datasets, both the head h and tail t of train triples have another semantically valid counterpart for negative sampling. In addition, all relations appearing in the training set have a defined domain and range. These two conditions guarantee that each positive train triple can be paired with at least one semantically valid triple. Consequently, the impact of our signature-driven loss functions can be assessed in a controlled experimental setting.

Baseline models. In these experiments, the following KGEMs are used: TransE, TransH, DistMult, ComplEx, SimplE, ConvE, TuckER, and RGCN. Details about these models can

Table 5.3: Datasets used in the experiments. These are filtered versions of the standard FB15k237, DB93k, and YAGO4-19k

Dataset	$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{R}\rvert$	$\lvert\mathcal{T}_{train}\rvert$	$\lvert\mathcal{T}_{valid}\rvert$	$\lvert\mathcal{T}_{test}\rvert$
FB15k187	14,305	187	245,350	15,256	17,830
DB77k	76,651	150	140,760	16,334	32,934
YAGO4-14k	14,178	37	18,263	472	448

be found in Section 2.4. Table 5.4 summarizes their main characteristics with regards to our experiments.

Table 5.4: Characteristics of the KGEMs used in the experiments

Model	Scoring Function	Loss Function
TransE [18]	$\lVert\mathbf{e}_h + \mathbf{e}_r - \mathbf{e}_t\rVert_p$	$\mathcal{L}_{PHL}$
TransH [174]	$\lVert\mathbf{e}_{h_\perp} + \mathbf{d}_r - \mathbf{e}_{t_\perp}\rVert_p$	$\mathcal{L}_{PHL}$
DistMult [181]	$\langle\mathbf{e}_h, \mathbf{W}_r, \mathbf{e}_t\rangle$	$\mathcal{L}_{PHL}$
ComplEx [164]	$\mathrm{Re}\left(\mathbf{e}_h \odot \mathbf{e}_r \odot \overline{\mathbf{e}}_t\right)$	$\mathcal{L}_{PLL}$
SimplE [93]	$\frac{1}{2}\left(\langle\mathbf{e}_h^h, \mathbf{e}_r, \mathbf{e}_t^t\rangle + \langle\mathbf{e}_h^t, \mathbf{e}_r^{-1}, \mathbf{e}_t^h\rangle\right)$	$\mathcal{L}_{PLL}$
ConvE [39]	$g\left(\mathrm{vec}\left(g\left(\mathrm{concat}\left(\widehat{\mathbf{e}}_h, \widehat{\mathbf{e}}_r\right) * \omega\right)\right)\mathbf{W}\right) \cdot \mathbf{e}_t$	$\mathcal{L}_{BCEL}$
TuckER [7]	$\mathcal{W} \times_1 \mathbf{e}_h \times_2 \mathbf{w}_r \times_3 \mathbf{e}_t$	$\mathcal{L}_{BCEL}$
RGCN [147]	DistMult decoder	$\mathcal{L}_{BCEL}$

Implementation and hyperparameters. For the sake of comparison, MRR, Hits@K and Sem@K are computed after training models from scratch. A maximum of 400 epochs of training was set, as in LibKGE [22]. RGCN is the only model that needed additional training due to lower convergence. Therefore, we trained it during 4,000 epochs. Except with ConvE, TuckER, and RCGN that implement 1-N scoring, UNS [18] was used to pair each train triple with two corresponding negative triples: one which is semantically invalid, and one which is semantically valid. Regarding BCEL, each positive triple is scored against negative triples formed with all other entities in the graph. Hence, negative triples comprise both semantically valid and invalid triples. In order to ensure a fair comparison between models, embeddings are initialized with the same seed and each model is fed with exactly the same set of negative triples at each epoch. Grid-search based on predefined hyperparameters was performed. The full hyperparameter space, as well as the optimal hyperparameters are reported in C.1 and C.2, respectively.

5.3.3 Results

Global Performance

Table B.3 displays KGEM performance, datasets and evaluation metrics of interest. It clearly shows that, with the sole exception of SimplE on DB77k, the signature-driven loss functions $\mathcal{L}^S_{PHL}$, $\mathcal{L}^S_{BCEL}$, and $\mathcal{L}^S_{PLL}$ all lead to significant improvement in terms of Sem@10. Importantly, in some cases the relative gain in Sem@10 compared to the corresponding vanilla loss function is huge (underlined in Table B.3): +137%, +117%, and +100% for RGCN, SimplE, and ComplEx on the smaller YAGO4-14k dataset, respectively, and +69% and +64% for ComplEx and SimplE

Table 5.5: Rank-based and semantic-based results. Bold fonts indicate which model performs best w.r.t. a given metric. Suffixes V and S indicate whether the model is trained under the vanilla or signature-driven version of the loss function, respectively. Hits@10 and Sem@10 are abbreviated to H@10 and S@10. Underlined cells indicate results that are more specifically referred to in Section 5.3.3

	FB15k187			DB77k			YAGO4-14k		
	MRR	H@10	S@10	MRR	H@10	S@10	MRR	H@10	S@10
TransE-V	.260	.446	.842	.274	.438	.936	.868	**.945**	.795
TransE-S	**.315**	**.497**	**.973**	**.275**	**.440**	**.985**	**.876**	.944	**.968**
TransH-V	.266	.450	.855	.270	.437	.907	.836	.944	.581
TransH-S	**.319**	**.501**	**.973**	**.274**	**.442**	**.980**	**.857**	**.945**	**.831**
DistMult-V	.291	.457	.824	.295	.405	.784	.904	**.930**	.409
DistMult-S	**.332**	**.504**	**.971**	**.300**	**.416**	**.901**	**.912**	.929	**.449**
ComplEx-V	.280	.416	.472	**.309**	**.415**	.769	**.925**	**.932**	.333
ComplEx-S	**.316**	**.476**	**.796**	.297	.409	**.897**	.923	.931	**.667**
SimplE-V	.261	.387	.462	**.259**	**.346**	**.883**	**.926**	**.931**	.355
SimplE-S	**.268**	**.409**	**.759**	.230	.302	.850	.924	.927	**.769**
ConvE-V	.273	.470	.973	.273	.382	.935	**.934**	**.942**	.904
ConvE-S	**.283**	**.476**	**.996**	**.283**	**.405**	**.985**	.933	.940	**.997**
TuckER-V	.316	.516	.985	.311	.410	.912	.923	.927	.781
TuckER-S	**.320**	**.522**	**.996**	**.312**	**.421**	**.969**	**.931**	**.943**	**.929**
RGCN-V	.241	.386	.775	.194	.297	.872	.911	.923	.349
RGCN-S	**.260**	**.415**	**.860**	**.197**	**.320**	**.957**	**.927**	**.934**	**.828**

on FB15k187, respectively. These gains in terms of Sem@10 are observed regardless of the loss function at hand, which demonstrates that the proposed signature-driven loss functions can all drive KGEM semantic correctness towards more satisfying results. It is worth noting that, in most cases, they also lead to better KGEM performance as measured by MRR and Hits@10. We observe that in 19 out of 24 ($\approx 79\%$) one-to-one comparisons between the same KGEM trained with vanilla vs. signature-driven loss functions, better MRR values are reported for the model trained under the signature-driven loss function. The remaining comparisons ($\approx 21\%$) highlight minimal losses in terms of MRR, often for the benefit of significantly better Sem@10 values. This observation raises the question whether better MRR and Hits@K should be pursued at any cost, or whether a small drop w.r.t. to these metrics is acceptable if this leads to a significantly better KGEM semantic correctness (see Section 5.3.4). Plus, these promising results imply that even if the intended purpose is to only maximize MRR and Hits@K values, taking the available signature information into consideration is strongly advised: this does not only improve KGEM semantic correctness, but also provide performance gains in terms of MRR and Hits@K.

In the following, we provide a finer-grained results' analysis, which focuses on the different loss functions and datasets used in the experiments. Although we previously showed the effectiveness of our approach, the benefits brought from considering BK in the form of relation domains and ranges differ across loss functions. In particular, the gains achieved using $\mathcal{L}^{S}_{PHL}$ and $\mathcal{L}^{S}_{BCEL}$ are substantial. With these loss functions, we observe a systematic improvement w.r.t. Sem@10.

Gains are also reported w.r.t. rank-based metrics in the vast majority of cases. The only exception is on YAGO4-14k where semantic losses are sometimes slightly outperformed but still competitive w.r.t. their vanilla counterparts. However, the difference in terms of MRR and Hits@10 values is negligible. Additionally, YAGO4-14k is a small dataset with a reduced number of triples. Hence, this result indicates that on small datasets, the additional signature information improves Sem@K but the resulting enhanced embedding space does not help discriminate gold entities from others. The scarcity of the dataset and of the resulting embedding space seems to suffice in this regard. Therefore, incorporating BK about relation domains and ranges into $\mathcal{L}^S_{PHL}$ and $\mathcal{L}^S_{BCEL}$ is a viable approach that provides consistent gains both in terms of MRR, Hits@10 and Sem@10. Regarding the benefits from doing so under $\mathcal{L}^S_{PLL}$, we can notice a slight decline in MRR and Hits@10 values in some cases. However, the other side of the coin is that achieved Sem@10 values are substantially higher: except for SimplE on DB77k, Sem@10 values increase in a range from +17% to +117% for the remaining one-to-one comparisons. As such, our approach using $\mathcal{L}^S_{PLL}$ also provides satisfactory results, as long as a slight drop in rank-based metrics is acceptable if it comes with the benefit of significantly better KGEM semantic correctness. Besides, it is worth noting the following points: our hyperparameter tuning strategy relied on the choice of the best ϵ value on YAGO4-14k – for computational limitations. The ϵ value which was found to perform the best on YAGO4-14k was subsequently used in all the remaining scenarios. A more thorough tuning of ϵ on the other datasets would have potentially provided even more satisfying results under the $\mathcal{L}^S_{PLL}$, thus strengthening the value of our approach.

Ablation Study

In this section, KGEMs are tested on three buckets of relations that feature narrow (B1), intermediate (B2), and large (B3) sets of semantically valid heads or tails, respectively. The cut-offs have been manually defined and are provided in supplementary materials (see C.5). The analysis of B1 allows us to gauge the impact of signature-driven loss functions on relations for which it is harder to predict semantically valid entities. Results reported in Table 5.6 for B1 clearly demonstrate that the impact of injecting BK into loss functions is exacerbated for them, thus supporting the value of our approach in a sparse and difficult setting. One might think that the better MRR values achieved with $\mathcal{L}^S_{PHL}$, $\mathcal{L}^S_{BCEL}$, and $\mathcal{L}^S_{PLL}$ are highly correlated to the better Sem@10 values. This is partially true, as for a relation in B1, placing all semantically valid candidates at the top of the ranking list is likely to uplift the rank of the ground-truth itself. However, we can see that RGCN-V and RGCN-S have almost equal MRR values on YAGO4-14k, while Sem@10 values of RGCN-S are much higher than RGCN-V (+254%). Similar findings hold for ComplEx on YAGO4-14k, ConvE on DB77k, TransE on DB77k and YAGO4-14k. This shows that in a number of cases, signature-driven loss functions improve the semantic correctness of KGEM for small relations while leaving its performance untouched in terms of rank-based metrics. Results on B2 and B3 are provided in supplementary materials. In particular, it can be noted that the relative benefit of our approach w.r.t. MRR and Hits@10 is more limited on such buckets. Regarding Sem@K, results achieved with the vanilla loss functions are already high, hence the relatively lower gain brought by injecting ontological BK. These already high Sem@K values may be explained by a higher number of semantically valid candidates.

5.3.4 Discussion

In this section, we dissect our experimental results to answer the previously introduced research question:

Table 5.6: Rank-based and semantic-based results on the bucket of relations that feature a narrow set of semantically valid heads or tails (B1). The cut-offs have been manually defined and are provided in supplementary materials. Bold fonts indicate which model performs best w.r.t. a given metric. Suffixes V and S indicate whether the model is trained under the vanilla or signature-driven version of the loss function, respectively. Hits@10 and Sem@10 are abbreviated to H@10 and S@10

	FB15k187			DB77k			YAGO4-14k		
	MRR	H@10	S@10	MRR	H@10	S@10	MRR	H@10	S@10
TransE-V	.535	.727	.647	.498	.578	.460	.914	**.979**	.620
TransE-S	**.646**	**.805**	**.937**	**.539**	**.626**	**.910**	**.922**	.975	**.924**
TransH-V	.541	.734	.661	.475	.555	.425	.897	.972	.436
TransH-S	**.655**	**.814**	**.936**	**.541**	**.643**	**.828**	**.923**	**.981**	**.684**
DistMult-V	.589	.735	.628	.466	.462	.244	.949	**.959**	.304
DistMult-S	**.667**	**.802**	**.929**	**.498**	**.547**	**.451**	**.965**	.958	**.372**
ComplEx-V	.530	.567	.116	**.424**	**.425**	.161	.955	.956	.133
ComplEx-S	**.637**	**.723**	**.537**	.421	.399	**.198**	**.961**	.956	**.423**
SimplE-V	.507	.553	.136	**.396**	**.370**	**.273**	**.959**	.882	**.932**
SimplE-S	**.576**	**.671**	**.505**	.324	.259	.206	.958	**.883**	.930
ConvE-V	.549	.779	.973	.518	**.569**	.789	.969	**.972**	.915
ConvE-S	**.562**	**.783**	**.986**	.518	.566	**.927**	.969	.965	**.960**
TuckER-V	.597	.811	.969	.519	.568	.740	.949	.970	.846
TuckER-S	**.598**	**.815**	**.973**	**.526**	**.582**	**.797**	**.964**	.970	**.892**
RGCN-V	.510	.629	.468	.386	.387	.254	.963	.959	.141
RGCN-S	**.549**	**.705**	**.682**	**.396**	**.415**	**.398**	**.966**	**.967**	**.499**

RQ3.2. How main loss functions used in LP can incorporate domain and range constraints to differently consider negative triples based on their semantic validity and what is their impact on the overall KGEM performance?

More specifically, we address **RQ3.2.** in two stages: first, we elaborate on the inclusion of domain and range constraints into the main loss functions. Second, we discuss the resulting impacts in terms of KGEM performance.

Inclusion of domain and range constraints into loss functions

The design of $\mathcal{L}^S_{PHL}$, $\mathcal{L}^S_{BCEL}$, and $\mathcal{L}^S_{PLL}$ is detailed in Section **??**. These loss functions provide adequate training objective for KGEMs, as evidenced in Table B.3. Most importantly, in Section **??** we clearly show how the inclusion of signature information into $\mathcal{L}^S_{PHL}$, $\mathcal{L}^S_{BCEL}$, and $\mathcal{L}^S_{PLL}$ can be brought under one roof thanks to a commonly defined semantic factor. Although this semantic factor operates at different levels depending on the loss function, its common purpose is to differentiate how semantically valid and semantically invalid negative triples should be considered compared to positive triples, whereas traditional approaches treat all negative triples indifferently. Besides, our approach which consists in transforming vanilla loss functions into signature-driven ones can also work for other loss functions such as the pointwise hinge loss or the pairwise logistic loss, as presented in [115]. Such losses can include a semantic factor ϵ as well. The tailoring of these losses and the experiments are left for future work.

Recall that compared to complex negative sampler such as KGBAN [27], NSCaching [186], and self-adversarial NS [157], we do not introduce any potential overhead due to the need for maintaining a cache [186] or training an intermediate adversarial learning framework [27, 157] for generating high-quality negatives. Instead, negative triples dynamically enter a different part of the loss function depending on their semantic validity. In future work, we will compare the performance and algorithmic complexity of our approach w.r.t. NS procedures, thereby highlighting the potential cost savings in computational resources and execution time compared to sophisticated NS. Our approach is also agnostic to the underlying NS procedure, and can work along with simple uniform random NS [18] as well as more complex procedures [27, 186, 157]. Besides, our approach can be applied even in the absence of BK. In this case, signature-driven loss functions reduce to their vanilla version.

Impact of signature-driven loss functions on performance

Mohamed *et al.* [114] investigate the effects of specific choices of loss functions on the scalability and performance of KGEMs w.r.t. rank-based metrics. In this present work, we also assess the semantic capabilities of such models. Based on the results provided and analyzed in Section **??**, incorporating BK about relation domains and ranges into the loss functions clearly contribute to a better KGEM semantic correctness, as evidenced by the huge increase frequently observed w.r.t. Sem@K.

It should also be noted that this increase is not homogeneously distributed across relations: relations with a smaller set of semantically valid entities as heads or tails are more challenging w.r.t. Sem@K (see Table 5.6 compared to results on B2 and B3 provided in supplementary materials). For such relations, the relative gain from signature-driven loss functions is more acute (*e.g.* Sem@10 results on B1). In addition, signature-driven loss functions also drive most of the KGEMs towards better MRR and Hits@K values. As shown in Table B.3, this is particularly the case when using the $\mathcal{L}^S_{PHL}$ and $\mathcal{L}^S_{BCEL}$. This result is particularly interesting, as it suggests that when semantic information about entities and relations is available, there are benefits in

using it, even if the intended goal remains to enhance KGEM performance w.r.t. rank-based metrics only.

In addition, it has been noted that including signature information in loss functions has the highest impact on small relations (B1) (Table 5.6), both in terms of rank-based metrics (MRR, Hits@10) and semantic correctness (Sem@10). For relations featuring a larger pool of semantically valid entities, the impact is still positive w.r.t. Sem@10, but it sometimes comes at the expense of a small drop in terms of rank-based metrics. If the latter metrics are the sole optimization objective, it would thus be reasonable to design an adaptive training strategy in which vanilla and signature-driven loss functions are alternatively used depending on the current relation and the number of semantically valid candidates as head or tail.

On the Evaluation of KGEM Predictions for LP

It should be noted that Sem@K measures the capability of models to predict semantically correct triples w.r.t. relation signatures and but does not measure falsehood, contrary to MRR or Hits@K. It can be argued that in specific use-cases, such as e-commerce recommender systems, models should predict the expected item (high MRR/Hits@K) but also predict semantically valid items (high Sem@K) for a good user experience. In other applications such as healthcare, aerospace, or security, users expect to notice when models are failing in order to be able to take over. This corresponds to high MRR/Hits@K and low Sem@K so that errors are clearly noticeable. Our experiments and these use cases illustrate the need of both Hits@K and Sem@K metrics to precisely qualify model performance w.r.t. the needs of the considered application. This opens perspectives considering distinct kinds of negative triples differently to tailor specific aspects of model performance to the requirements of the application.

5.4 Generating versatile embeddings for several tasks

The ultimate goal of KGEMs is to learn the optimal representation space for entity and relation embeddings. Obviously, obtaining the best possible embeddings is a computationally challenging task [128]. As a result, a well-informed initialization of these embeddings should accelerate convergence towards the optimal model parameters. Therefore, the focus of this section is on embeddings' initialization using schema-based information.

In the general field of graph representation learning, a few works highlighted the importance of embeddings' initialization. In [104] a graph partitioning technique is used to divide the graph into multiple disjoint subsets. Subsequently, an abstract graph based on these partitions is created. To initialize the embeddings for each node in the graph, the authors propose to compute the network embeddings on this smaller abstract graph, and then propagate these embeddings among the nodes in the original input graph. In [46], Fahrbach *et al.* propose a nearly-linear time algorithm that generates a coarsened graph on a subset of relevant nodes. Their experiments demonstrate that computing embeddings on the coarsened graph rather than on the full graph significantly reduces training time without penalizing accuracy.

Regarding KGs more specifically, Pietrasik *et al.* [134] present a strategy for coarsening KGs to improve embedding efficiency and quality. This process involves reducing the graph size by probabilistically collapsing similar entities while preserving structural and semantic properties. Their strategy includes three key steps: probabilistic graph coarsening, coarse graph embedding, and reverse mapping with fine-tuning. The results indicate significant improvements in most cases. However, their approach solely relies on probabilistic techniques and does not incorporate schema-based information. Besides, their approach is only benchmarked w.r.t. the node

classification task.

Even though this section follows a similar research direction, the approach we will present differs significantly from what has been explored so far. In this case, we will investigate how initializing the embeddings using schema-based information is possible, particularly with leveraging knowledge about entity types. Furthermore, while Pietrasik *et al.* [134] focused on node classification, and even though we have been primarily concerned with LP up to this point, this final section aims to be more general and addresses both of the aforementioned tasks, as well as a third one: entity clustering.

Consequently, we seek to answer the previously defined research question **RQ3.3**:

> **RQ3.3.** How to use class-based information to efficiently initialize entity and relation embeddings and how profitable is it for KG-based tasks?

5.4.1 Proposed approach

All of the aforementioned approaches use statistics over the graphs instead of considering schema information. In contrast, our work focuses on leveraging ontological constraints to generate an abstraction of the KG that encapsulates general knowledge about how entities and relations interact based on entity types, and relation domains and ranges. We name *protograph* such an abstraction. The general process of building a protograph is depicted in Fig. 5.2.

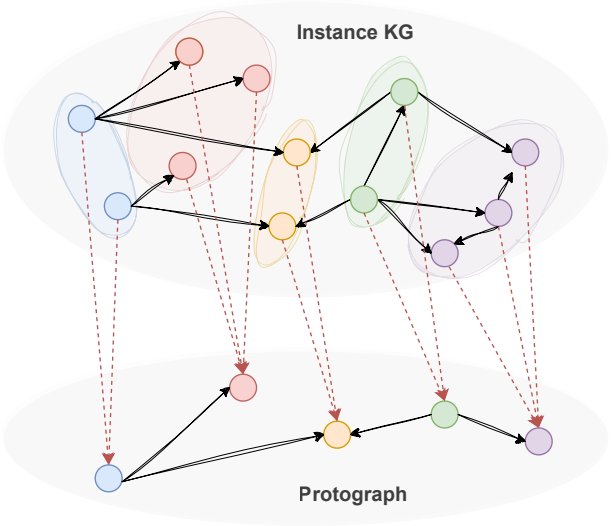


Figure 5.2: Mapping a KG to its protograph

If considering schema-based information is usually done with the goal of improving results w.r.t. a given task such as LP [35, 111], it would be worth investigating whether the learnt embeddings actually have stronger semantic capabilities [86, 80]. If so, they are expected to provide better performance with regards to other tasks such as entity classification and clustering.

In this work, we propose an approach named MASCHInE (Fig. 5.3) that leverages schema information in a model-agnostic way, making it applicable to arbitrary KGEMs. Specifically, we create and embed *protographs* from a KG schema. The protograph embeddings are then used to initialize the actual KGEs, and training starts on these pre-trained KGEs. Finally, we assess the versatility of these KGEs by comparing results with a vanilla, traditional learning approach.

First, we detail the two heuristics we propose for building protographs using schema-based information. Next, we provide some perspective on the overall proposed approach MASCHInE.

Protographs

Recall that most KGs are underpinned by a schema from which background knowledge about entities and relations can be extracted. The protographs we build leverage RDF/S information and contain an entity p_C for each class C, and the same set of relations as the KG. The first design principle consists in adding an edge for each relation with domain and range restrictions, *i.e.*, given the two axioms

$$domain(r, C_i),\ range(r, C_j) \tag{5.3}$$

we add a triple

$$(p_{C_i}, r, p_{C_j}) \tag{5.4}$$

to the protograph[25]. We refer to this strategy as **P1**.

The protograph P1 contains as many triples as there are relations with both a domain and range[26] in the KG. This may lead to relatively small-sized protographs from which there is little data for the KGEM to learn from. In addition, relation domains and ranges can refer to generic classes. If entities are typed in a fine-grained way – *i.e.* with more specific classes – such classes will be absent from the protograph. As such, we propose a second design principle. Assuming we observe the following axioms:

$$domain(r, C_i),\ range(r, C_j), \tag{5.5}$$

$$subclassOf(C'_i, C_i),\ subclassOf(C'_j, C_j) \tag{5.6}$$

we add the following triples to the protograph:

$$(p_{C_i}, r, p_{C_j}),\ (p_{C'_i}, r, p_{C_j}),\ (p_{C_i}, r, p_{C'_j}) \tag{5.7}$$

In other words: for each subclass of the class appearing in the domain or range of a relation, an additional triple is created.

We refer to this protograph as **P2**. Fig. 5.4 illustrates this process for a triple extracted from DBpedia.

P2 usually contains more triples than P1, at the possible expense of containing dubious ones. To illustrate, the DBpedia ontology contains the triples:

$$domain(currentWorldChampion, Sport) \tag{5.8}$$

$$range(currentWorldChampion, Agent) \tag{5.9}$$

$$subclassOf(Family, Agent) \tag{5.10}$$

which will lead to the triple

$$(P_{Sport}, currentWorldChampion, P_{Family}) \tag{5.11}$$

for which the relevancy and correctness are questionable. Such triples are still added to P2 and cannot be automatically ruled out. Conceptually, with P2, we accept some noise to

[25]Note that C_i and C_j might be the same, so there can be cycles in the protograph.

[26]In the KGs used in this paper, all relations have an explicit domain and range.

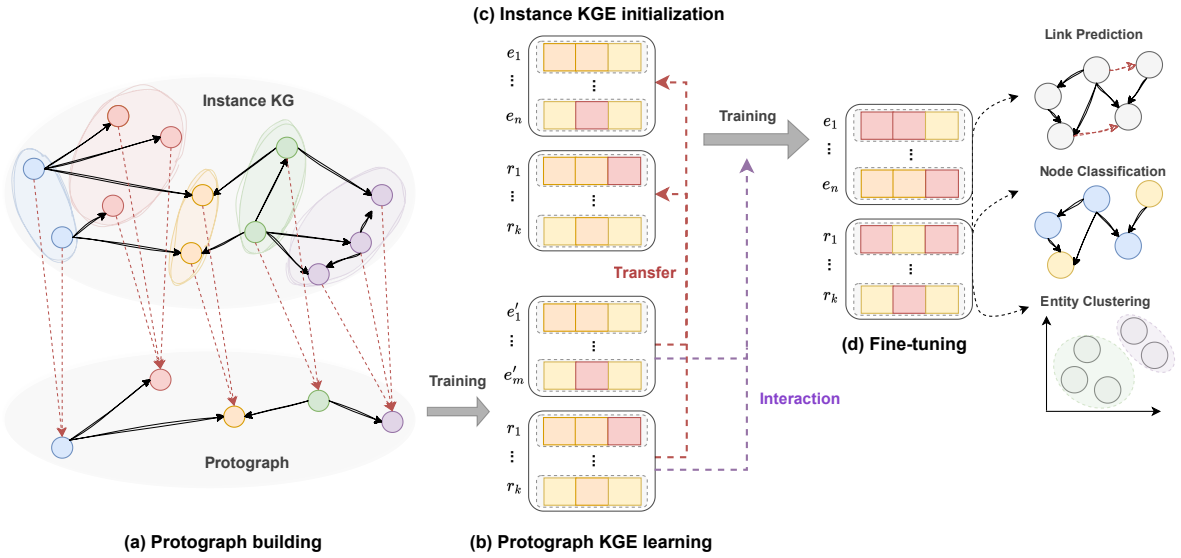


Figure 5.3: Overview of MASCHInE. The overall training approach is based on several modules that each features different possibilities. First, the protograph is built following predefined design principles (a). Protograph embeddings are learnt (b) before being transferred to the corresponding entities and relations of the KG (c). Next, training starts on the KG (d). During this step, embeddings learnt on the protograph can still intervene in the fine-tuning procedure (purple dotted arrow). Finally, the fine-tuned KGEs can be used in several tasks

be introduced in the protograph for the benefit of more triples being generated. However, the generation of dubious triples is limited by two factors: firstly, when adding triples that derive from an original triple (*i.e.* containing the domain and range of the relation), we only consider direct subclasses – in contrast to transitive subclasses. Secondly, we fix one side of the triple and generate as many new triples as there are direct subclasses of the class of the other side, i.e., we avoid generating triples $(p_{C'_i}, r, p_{C'_j})$.

MASCHInE

The proposed MASCHInE framework allows for generating pre-trained embeddings based on a protograph, to be further fine-tuned on the initial KG. MASCHInE is intended to produce more versatile KGEs that can be successfully used in various tasks. Conceptually, MASCHInE is made of four different modules that are depicted in Fig. 5.3 and further detailed below.

(a) Protograph building is the preliminary step that generates a second source of data to learn from. In particular, schema-based information is leveraged to create a protograph that encapsulates more general knowledge about how entities and relations interact based on ontological constraints such as entity types, relation domains and ranges, etc. A mapping dictionary is created to pair entities between the KG and the protograph (the set of relations in the KG and the protograph are identical). In this work, each KG entity maps to its (potentially multiple) most specific classes – represented as entities in the protograph. In Section 5.4.1, we presented two heuristics for building protographs with such constraints. Other possibilities exist and we leave them for future work.

(b) Protograph KGE learning is the task of learning KGEs on the protograph itself. This

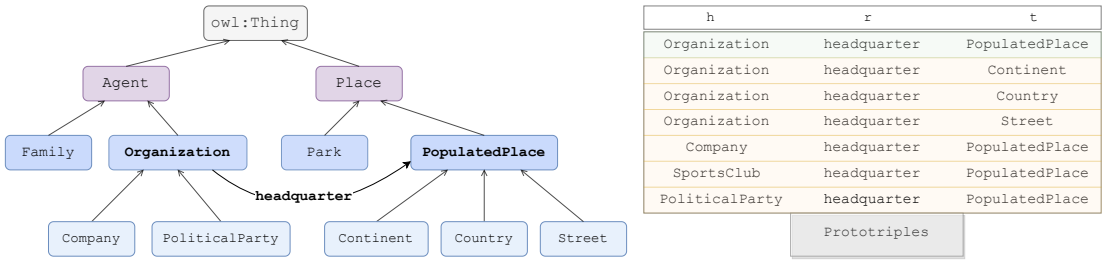


Figure 5.4: Excerpt from DBpedia class hierarchy (left) and the generated prototriples (right) according to P2. The full URI names have been shortened due to space limitations. The prototriple highlighted in green (right) comes from the schema, whereas the following triples are generated by fixing the domain (resp. range) of `dbo:headquarter` and replacing the class in the range (resp. domain) with each of its direct subclasses

can be achieved regardless of the KGEM at hand and is therefore model-agnostic[27].

(c) Instance KGE initialization consists in transferring protograph embeddings to the corresponding entity and relation embeddings of the KG. To do so, we assign each entity in the KG the embedding vector of its corresponding protograph entity using the dictionary created in step (a) (*i.e.*, its most specific class). When an entity maps to multiple protograph entities (*i.e.* when there exist mutiple most specific classes for a given entity), the protograph embeddings of its most specific classes are averaged and the resulting embedding is transferred to the entity.

(d) Fine-tuning is performed on the entity KGEs. At this stage, embeddings learnt on the protograph could still intervene in the fine-tuning procedure. In this work, however, we stick to plain entity KGE initialization, *i.e.* protograph embeddings are frozen after being transferred, and do not interact with entity KGEs anymore.

5.4.2 Results

In our experiments, we applied the MASCHInE framework we previously presented. Five popular KGEMs: TransE [18], DistMult [181], ComplEx [164], ConvE [39], and TuckER [7]. We first train them on protographs, and then use the protograph embeddings to initialize entity and relation embeddings in the KG. Training resumes on the original KG and the fine-tuned KGEs are ultimately assessed in three tasks, namely link prediction, entity clustering, and node classification. Importantly, the same KGEM is used on the protograph and the original KG. For each task, we compare the vanilla setting (V), *i.e.*, directly training the respective KGEM on the KG, with the MASCHInE approach based on P1 (resp. P2).

Datasets, optimal hyperparameters, scripts, and pre-trained embeddings are publicly released[28].

In the following, we assess our proposed learning approach based on the MASCHInE framework w.r.t. three tasks: link prediction (LP), entity clustering (EC), and node classification (NC).

[27] In our experiments, the same KGEM is used for both pre-training over the protograph and fine-tuning embeddings on the KG.

[28] https://github.com/nicolas-hbt/versatile-embeddings

Table 5.7: KG and protograph characteristics. Column headers from left to right: number of entities, relations, and triples

Dataset		$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{R}\rvert$	$\lvert\mathcal{T}\rvert$
YAGO4-14k	KG	14,178	37	19,183
	P1	22	37	37
	P2	590	37	4,959
FB15k187	KG	14,305	187	278,436
	P1	138	187	187
	P2	138	187	187
DB77k	KG	76,651	150	190,028
	P1	55	150	150
	P2	186	150	3,210

Link prediction

Datasets. Due to the need for schema-defined KGs, the following three benchmark datasets are considered: YAGO4-14k, FB15k187, and DB77k [80]. In particular, all entities are typed and relations have both a defined domain and range. Table 5.7 provides statistics for these datasets and their respective protographs P1 and P2. Note that protograph entities are classes from the original KG. Full dataset description is provided at: `https://github.com/nicolas-hbt/versatile-embeddings#datasets`.

Evaluation metrics. This section is motivated by evaluating the fine-tuned KGEs for LP. Performance is assessed w.r.t. Mean Reciprocal Rank (MRR), Hits@K, and Sem@K.

Implementation details. KGEMs were implemented in PyTorch. KGEMs are trained during 400 epochs. For TransE, DistMult, and ComplEx, UNS [18] was used to pair each train triple with one negative counterpart. ConvE and TuckER are trained using 1-N scoring [39]. In this work, we stick with the loss function originally used for each KGEM. Evaluation is performed every 10 epochs and the best model w.r.t. MRR on the validation set is saved for performance assessment on the test set. The aforementioned procedure prevails for the vanilla training mode. For P1 and P2, we use the same procedure as for training on the KG, except that 200 epochs of training are previously performed on the protograph, for a total of 600 training epochs.

Experimental results. LP results are reported in Table 5.8. On YAGO4-14k and DB77k, learning embeddings with P1 or P2 provides better Sem@K values compared to the case where no protograph is used, *i.e.* tha vanilla case (V). No significant change is noticed on FB15k187. This is likely to be an artefact of the Freebase class hierarchy and how entities are typed in FB15k187: the class hierarchy only has 2 levels, all relation domains and ranges are level-2 classes (therefore P1 and P2 are the same, see Table 5.7) and entities share many classes together. In this situation, the protograph may not contain discriminative enough information for pre-training embeddings, which is reflected in the similar results achieved regardless of the training setting. In addition, we observe that some KGEMs seem to benefit more from the MASCHInE approach. In particular, ConvE and TuckER tend to perform better w.r.t. both rank-based and semantic-oriented metrics. Overall, we notice that including protographs in the training phase helps making semantically valid predictions, as evidenced by the better Sem@K

under P1 and P2. Protographs may help organize the embedding space in such a way that geometric distances between embeddings better reflect their semantic similarities. Regarding rank-based metrics, we notice equivalent performance, which could be explained since LP relies less on semantics and more on relational patterns [11] compared to other tasks such as entity clustering (see below).

Table 5.8: Link prediction results on YAGO4-14k, FB15k187, and DB77k using KGEMs trained without protograph (V), with P1, and P2. For a given dataset and KGEM, comparisons are made between the three settings, and best results are in bold fonts. Hits@K and Sem@K are abbreviated as H@K and S@K, respectively

		YAGO4-14k					FB15k187					DB77k				
		MRR	H@3	H@10	S@3	S@10	MRR	H@3	H@10	S@3	S@10	MRR	H@3	H@10	S@3	S@10
TransE	V	**0.766**	**0.806**	**0.885**	0.987	0.971	**0.232**	**0.259**	**0.430**	**0.970**	0.958	**0.236**	**0.285**	**0.405**	0.983	0.976
	P1	0.746	0.791	0.871	0.994	0.993	0.227	0.254	0.425	**0.970**	**0.959**	0.217	0.260	0.382	0.985	0.980
	P2	0.700	0.762	0.854	**1.000**	**0.999**	0.227	0.254	0.425	**0.970**	**0.959**	0.216	0.260	0.380	**0.987**	**0.982**
DistMult	V	**0.899**	**0.916**	**0.926**	0.658	0.490	**0.228**	**0.256**	**0.412**	0.985	0.969	**0.282**	**0.308**	**0.391**	0.845	0.804
	P1	0.164	0.169	0.314	**1.000**	**1.000**	0.206	0.228	0.394	**0.988**	**0.980**	0.194	0.219	0.303	0.928	0.939
	P2	0.289	0.325	0.511	0.999	0.999	0.206	0.228	0.394	**0.988**	**0.980**	0.190	0.207	0.287	**0.937**	**0.947**
ComplEx	V	**0.923**	**0.930**	0.933	0.723	0.587	**0.203**	**0.223**	**0.399**	**0.942**	**0.923**	**0.286**	**0.317**	**0.394**	0.823	0.754
	P1	0.897	0.916	0.932	0.907	0.848	0.185	0.201	0.371	0.922	0.900	0.209	0.226	0.302	0.856	0.839
	P2	0.914	**0.930**	**0.934**	**0.914**	**0.854**	0.185	0.201	0.371	0.922	0.900	0.209	0.225	0.306	**0.913**	**0.890**
ConvE	V	**0.934**	**0.936**	**0.940**	0.858	0.824	0.265	0.297	0.463	**0.979**	**0.971**	0.227	**0.259**	0.352	0.897	0.889
	P1	0.927	0.930	0.935	0.882	0.845	0.272	0.301	**0.470**	0.977	0.970	0.227	0.256	0.352	0.932	0.932
	P2	0.931	0.931	0.938	**0.935**	**0.922**	**0.272**	**0.301**	**0.470**	0.977	**0.971**	**0.229**	**0.259**	**0.356**	**0.943**	**0.942**
TuckER	V	0.919	0.931	**0.942**	0.899	0.821	0.282	0.311	0.467	**0.996**	**0.991**	0.274	0.305	0.401	0.988	0.985
	P1	**0.922**	**0.933**	0.941	0.951	0.925	**0.286**	**0.317**	**0.472**	0.995	0.985	**0.277**	**0.308**	0.402	0.987	0.984
	P2	0.918	0.930	0.941	**0.971**	**0.939**	**0.286**	**0.317**	**0.472**	0.995	0.985	0.275	0.304	**0.403**	**0.989**	**0.986**

Entity clustering

Jain *et al.* [86] demonstrate that embeddings do not consistently capture the semantics of the KG. However, it is worth noting that they only considered schema-agnostic approaches. In this section, we investigate whether embeddings learnt for the LP task under P1 and P2 can actually prove useful in EC, therefore investigating how injecting schema-based information during embeddings' initialization impacts the overall organization of the resulting embedding space.

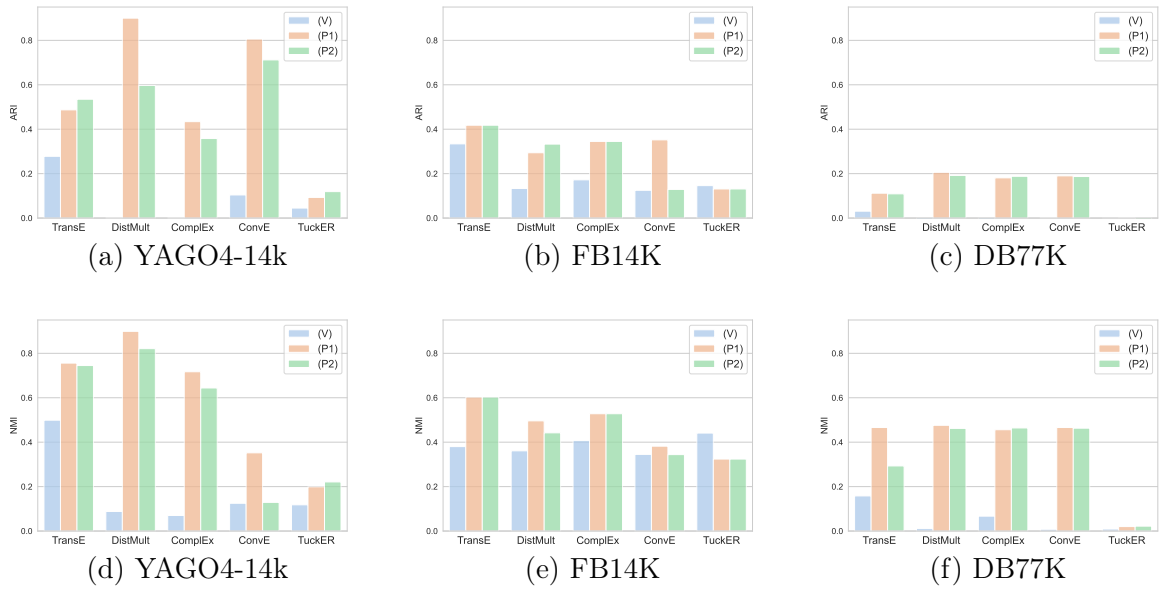
Datasets. We reuse YAGO4-14k, FB15k187, and DB77k as previously presented. The ground-truth labels are the most-generic classes an entity belongs to. It should be noted that entities belonging to several most-generic classes are filtered out as they would be assigned multiple ground-truth labels.

Implementation details. Entity embeddings are retrieved at the best epoch on the validation sets of YAGO4-14k, FB15k187, and DB77k, for all the KGEMs and under the three settings V, P1, and P2. Then, following the evaluation protocol in Jain *et al.* [86], the k-means clustering algorithm is run using default parameters of scikit-learn[29].

Evaluation metrics. Clustering results are evaluated using the Adjusted Rand Index (ARI) – which measures the similarity between the cluster assignments by making pairwise comparisons – and Normalized Mutual Information (NMI) – which measures the agreement between the cluster assignments. It should be noted that these metrics are applicable as we have the ground-truth labels for each entity. In particular, we compare clusters output by k-means with ground-truth labels that are the classes entities belong to. ARI and NMI results are reported in Fig. 5.5. Lowest and highest scores are 0 (-1 for ARI) and 1, respectively.

Experimental results. Except for TuckER on FB15k187, both settings P1 and P2 provide substantial improvements. It seems that the entity embeddings learnt by TuckER are not able to cluster entities based on their classes, which diminishes the relevancy of using this model for drawing comparisons. That being said, the benefit of P1 and P2 is consistent across datasets. This is even more striking for YAGO4-14k, as evidenced in Fig. 5.6: entities sharing the same ground-truth class tend to cluster more under P2. Jain *et al.* [86] pointed out the limitations of using KGEs for semantic representation of the underlying entities and relations. The relatively low performance of embeddings for EC under V (Fig. 5.5 and Fig. 5.6) indeed suggests that the semantic similarity between entities is not reflected through geometric similarities in their embeddings. However, P1 and P2 generate entity latent representations whose geometric similarities are more representative of their classes.

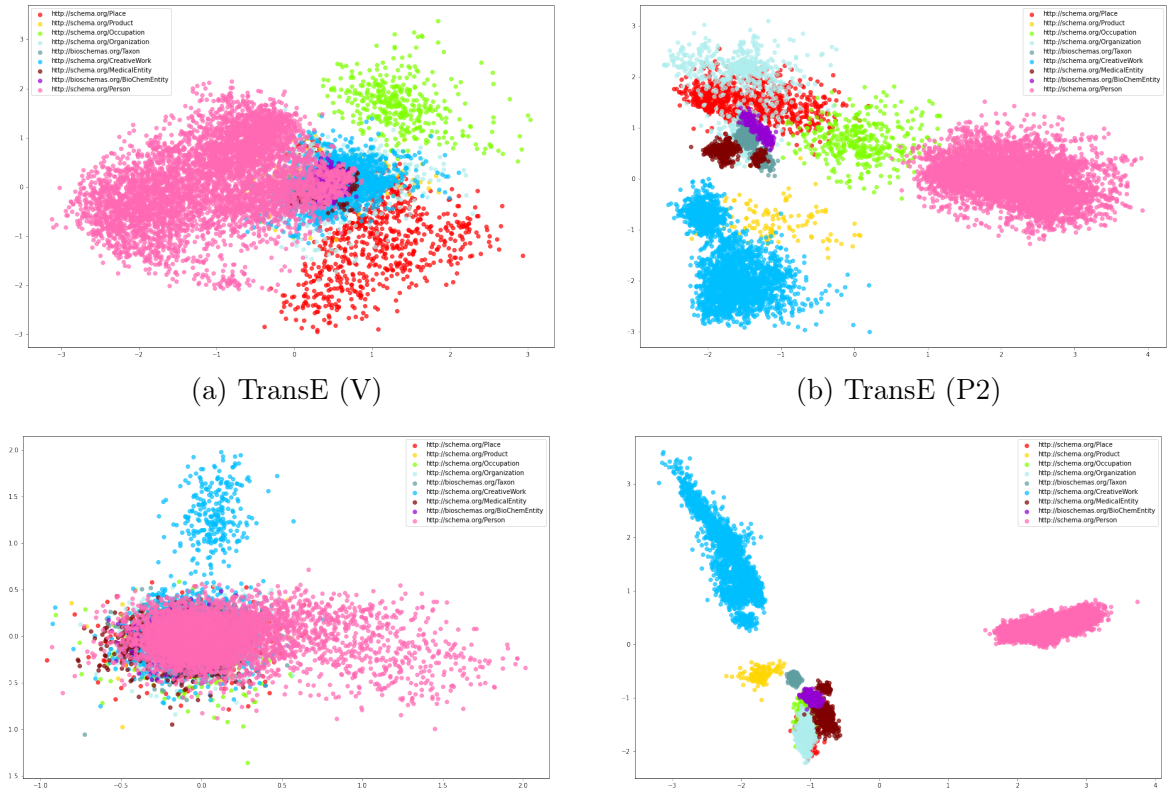
[29] `https://scikit-learn.org/stable/index.html`



(a) YAGO4-14k (b) FB14K (c) DB77K

(d) YAGO4-14k (e) FB14K (f) DB77K

Figure 5.5: ARI and NMI results on YAGO4-14k, FB14K, and DB77K, for each combination of model and setting



(a) TransE (V) (b) TransE (P2)

(c) DistMult (V) (d) DistMult (P2)

Figure 5.6: PCA visualizations of YAGO4-14k entity embeddings using the first two principal components. Each point represents an entity, whose color denotes its ground-truth class

Table 5.9: Entity clustering results on the GEval datasets. Bold font indicates which setting performs the best for a given model and dataset, while underlined results correspond to the second best setting

		CC						CCB						CAMAP					
		ARI	AMI	VM	FM	H	C	ARI	AMI	VM	FM	H	C	ARI	AMI	VM	FM	H	C
TransE	V	0.179	0.140	0.141	0.596	0.143	0.139	0.183	0.147	0.147	0.649	0.164	0.133	0.564	0.783	0.785	0.736	0.951	0.668
	P1	0.621	0.514	0.515	0.823	0.505	0.526	0.288	0.311	0.311	0.692	0.349	0.280	0.593	0.795	0.796	0.756	0.956	0.682
	P2	**0.680**	**0.570**	**0.571**	**0.849**	**0.563**	**0.579**	**0.299**	**0.314**	**0.314**	**0.698**	**0.353**	**0.284**	**0.646**	**0.815**	**0.816**	**0.791**	**0.964**	**0.708**
DistMult	V	0.018	0.024	0.027	0.709	0.015	0.095	−0.024	0.009	0.010	0.770	0.006	0.028	−0.054	0.099	0.106	0.478	0.087	0.136
	P1	**0.891**	0.817	0.817	**0.948**	0.813	0.821	**0.762**	**0.673**	**0.673**	**0.907**	**0.708**	**0.642**	0.655	0.705	0.707	0.798	0.782	0.646
	P2	**0.891**	**0.824**	**0.824**	**0.948**	**0.818**	**0.830**	0.756	0.626	0.627	**0.907**	0.646	0.608	**0.933**	**0.872**	**0.872**	**0.963**	**0.886**	**0.859**
ComplEx	V	−0.011	0.009	0.011	0.694	0.007	0.033	−0.024	0.009	0.010	0.771	0.006	0.028	0.139	0.345	0.350	0.520	0.341	0.360
	P1	**0.897**	**0.827**	**0.828**	**0.951**	**0.823**	**0.832**	**0.619**	**0.560**	**0.560**	**0.845**	**0.606**	**0.521**	**0.969**	**0.916**	**0.917**	**0.983**	**0.939**	**0.896**
	P2	0.319	0.264	0.265	0.665	0.269	0.262	0.493	0.460	0.460	0.788	0.507	0.422	0.961	0.914	0.914	0.978	0.938	0.892
ConvE	V	0.311	0.234	0.235	0.691	0.223	0.249	0.271	0.242	0.243	0.687	0.271	0.220	0.591	0.608	0.611	0.76	0.672	0.559
	P1	0.878	0.805	0.805	0.942	0.799	0.811	0.719	0.628	0.628	0.889	0.665	0.596	0.934	0.880	0.881	0.964	0.928	0.839
	P2	**0.916**	**0.848**	**0.848**	**0.960**	**0.847**	**0.849**	**0.790**	**0.697**	**0.697**	**0.919**	**0.727**	**0.669**	**0.954**	**0.915**	**0.915**	**0.975**	**0.968**	**0.868**
TuckER	V	**0.179**	**0.138**	**0.139**	**0.596**	**0.141**	**0.137**	**0.158**	**0.120**	**0.121**	**0.639**	**0.135**	**0.109**	**0.396**	0.526	0.529	**0.618**	0.636	0.453
	P1	0.069	0.051	0.052	0.543	0.053	0.051	0.137	0.094	0.094	0.631	0.105	0.086	0.376	**0.635**	**0.637**	0.599	**0.795**	**0.531**
	P2	0.047	0.042	0.043	0.533	0.043	0.042	0.033	0.051	0.052	0.578	0.058	0.046	0.276	0.487	0.490	0.520	0.621	0.405

Experiments on GEval clustering problems

In this follow-up entity clustering experiment, the GEval evaluation framework [130] is used. GEval provides gold standard datasets based on DBpedia and suggest clustering models, configurations and evaluation metrics[30].

Datasets. GEval proposes 4 datasets based on DBpedia. The first dataset contains entities of classes `dbo:AmericanFootballTeam` and `dbo:BasketballTeam`. We do not use this dataset as DB77k does not contain any of these entities. Instead, we use the other three datasets for which DB77k has a biggest coverage. For clarity, we name these datasets as CC, CCB, and CAMAP. The first two ones contain different numbers of DBpedia entities of classes `dbo:Cities` and `dbo:Countries`, while the third dataset contains five types of entities. Therefore, there are two ground-truth clusters in the first two datasets and five clusters in the third one.

Implementation details. We rely on GEval guidelines and perform the preliminary filtering step described as follows. First, we select entity embeddings learnt on DB77k for the LP task (see Section **??**). Embeddings of entities that do not appear in the GEval benchmark datasets are filtered out. Clustering with all the algorithms suggested in the GEval framework is performed on the remaining entities and evaluated w.r.t. several metrics.

Evaluation metrics. As in the previous experiment, we use the ARI. Following the GEval guidelines, we additionally use Adjusted Mutual Information Score (AMI), V-Measure (VM), Fowlkes-Mallow Score (FM), Homogeneity (H), and Completeness (C). V-Measure is the harmonic mean of homogeneity and completeness measure, while the Fowlkes-Mallow Score is the geometric mean of pairwise precision and recall. All these metrics measure the correctness of the cluster assignments based on ground-truth labels. For each of these datasets, and under each setting, results achieved with the best performing clustering method are shown in Table 5.9. Lowest and highest scores are 0 (-1 for ARI) and 1, respectively.

Experimental results. First, it is worth noting that results on CAMAP are significantly higher than on the other two datasets. Recall that CAMAP includes entities belonging to five different classes, whereas CC and CCB only contain entities from `dbo:Cities` and `dbo:Countries`. In this experiment, the embeddings learnt by the KGEMs are most useful for performing EC with five ground-truth clusters than with two, which might seem counterintuitive. However, in [86], it is demonstrated that in a KG, only a small subset of its entities actually have an embedding with a meaningful semantic representation. This is the reason why for the clustering task, results are not consistent across subsets of entities [86]. This is in line with our experimental findings that lead to the same conclusion.

[191] is also concerned with the ability of KGEMs to embed entities from the same class to the same region of the embedding space. Their results suggest that entities belonging to rare classes have embeddings that are less separable w.r.t. to entity embeddings of other classes compared with entities belonging to frequent classes. This assumption needs to be qualified in light of our experimental findings: lots of entities in DB77k are `dbo:City` and `dbo:Country`. Following [191], we should expect better results for CC and CCB in Table 5.9. However, it should be noted that both `dbo:City` and `dbo:Country` are subclasses of `dbo:Place` (equivalent to `dbo:Location`), which is over-represented in DB77k[31]. It is arguably harder to separate entity embeddings for

[30] `https://github.com/mariaangelapellegrino/Evaluation-Framework`
[31] In DB77k, 11,586 entities belong to the `dbo:Place` class.

CC and CCB, as all of them actually belong to the same parent class. In contrast, CAMAP contain albums, cities, companies, movies, and universities. In this case, `dbo:City` is a child class of `dbo:Place`, `dbo:Company` and `dbo:University` are child classes of `dbo:Agent` while `dbo:Album` and `dbo:Movie` are child classes of `dbo:Work`. With three different generic classes, the semantic similarities between entities in CAMAP is more easily reflected into the geometric similarities of their embeddings.

Interestingly, the discrepancies in results, *i.e.*, better results on CAMAP compared with CC and CCB, are setting-dependent (V, P1, P2). Indeed, they are alleviated when using either P1 or P2. Under V, TransE, ComplEx, and ConvE provide substantially better results on CAMAP compared to CC. Once using P2, we cannot see any significant difference in results on CAMAP compared to CC for these models. This means that training with a protograph-assisted approach led the KGEMs to generate higher-quality embeddings in terms of class separability, especially for entities in CC and CCB that are harder to distinguish. With the sole exception of TuckER whose performance may be due to non-optimal hyperparameter tuning, strong improvements are achieved w.r.t. the EC task on all datasets. Interestingly, not only V is outperformed by either P1 or P2 in each scenario, but actually by both of them at the same time. As a result, our schema-based, protograph-assisted learning approach MASCHInE proves effective for EC, regardless of the protograph design principle.

Node classification

Following on from the previous clustering experiment that clearly showed the benefit of incorporating pre-trained protograph embeddings, in this section we are interested in performing node classification with such embeddings. In this experiment, entities are labelled on the basis of whether they comply with some description logic (DL) constructors.

Datasets. In this experiment, we use the synthetic DLCC node classification benchmark [138]. The DLCC benchmark helps analyze the quality of KGEs by evaluating them w.r.t. which kinds of DL constructors they can represent. We use the synthetic gold standard datasets[32]. These are 12 synthetic datasets featuring labelled entities to classify and underpinned by a schema defining entity types, relation domains and ranges. The synthetic part of DLCC has been chosen because the authors argue that those pose classification problems which are not influenced by any side effects and correlations in the knowledge graph.

Implementation details. For these 12 test cases, their respective protographs are built. Then, the KGEMs are trained w.r.t. the LP task under the three different settings V, P1, and P2. In line with [138], multiple classifiers were trained for each test case: decision trees (DT), naive bayes (NB), k-nearest neighbors (k-NN), support vector machine (SVM), and multilayer perceptron (MLP). Based on the best entity embeddings learnt during LP, these models are used to classify entities using the default scikit-learn parameters.

Evaluation metrics. NC performance is measured using F-score, which is the harmonic mean of precision and recall. Lowest and highest scores are 0 and 1, respectively. For each model and test case, results achieved with the best classifier are reported in Table 5.10.

Experimental results. First, not all of the gold standard test cases are equally hard for the

[32] `https://github.com/janothan/DL-TC-Generator`

task of NC. For instance, entities in tc01, tc05, and tc06 are obviously easier to label than in the other datasets. Particularly, the DLCC synthetic gold standard datasets are built on the basis of different DL constructors in order to analyze which KG constructs are more easily recognized by KGEMs.

For the less trivial cases, the gains that can be achieved by using protographs are more significant. Especially in cases which are fairly badly solved by vanilla embeddings (*e.g.*, tc07, tc12), there are gains of more than 10 percentage points in F1. It is further remarkable that on this problem, compared to link prediction and entity clustering, TuckER can also benefit from the protographs.

When looking at which protograph construction strategy is more suitable for which kind of problem, we see that for those test cases which use a concrete class in their definition (particularly tc07, tc08, tc11, and tc12), P2 is clearly superior to P1. This may be due to the fact that P1 does not create class embeddings for all classes used in the respective constructors. Therefore, the resulting entity embeddings are less discriminating.

Table 5.10: Node classification results on the 12 DLCC test cases. Listed are the results of the best classifier for each combination of model and test case. Bold fonts indicate which setting performs the best for a given model and test case. Underlined results show which combination of model and setting performs the best for each test case.

Test Case	Setting	TransE	DistMult	ComplEx	ConvE	TuckER	Test Case	Setting	TransE	DistMult	ComplEx	ConvE	TuckER
	V	**0.702**	**0.810**	**0.820**	**0.885**	0.832		V	**0.654**	0.564	0.566	0.732	0.669
tc01	P1	0.647	0.754	0.784	0.875	**0.857**	tc02	P1	0.566	0.554	0.554	**0.764**	0.637
	P2	0.637	0.794	0.769	0.852	0.792		P2	0.571	**0.627**	**0.627**	0.687	**0.679**
	V	0.584	**0.571**	**0.554**	0.629	**0.669**		V	0.638	0.540	0.512	0.818	0.570
tc03	P1	**0.589**	0.541	0.519	**0.699**	0.664	tc04	P1	0.760	**0.812**	**0.832**	**0.870**	0.552
	P2	0.521	0.544	0.526	0.622	0.556		P2	**0.805**	0.802	0.818	0.818	**0.680**
	V	0.704	0.683	0.678	0.822	0.814		V	0.645	**0.940**	**0.938**	**0.940**	**0.950**
tc05	P1	0.887	0.937	**0.940**	**0.940**	0.814	tc06	P1	0.718	0.875	0.898	0.935	**0.950**
	P2	**0.937**	**0.940**	**0.940**	**0.940**	**0.902**		P2	**0.740**	0.828	0.788	0.895	0.942
	V	0.578	0.515	0.552	0.560	0.522		V	0.552	0.552	0.520	**0.598**	0.575
tc07	P1	0.505	0.540	0.560	0.560	0.552	tc08	P1	0.585	0.560	0.558	0.575	0.580
	P2	**0.582**	**0.605**	**0.655**	**0.675**	**0.648**		P2	**0.595**	**0.588**	**0.588**	0.578	**0.672**
	V	0.552	0.522	0.525	0.730	0.745		V	0.578	**0.572**	**0.562**	**0.640**	0.690
tc09	P1	**0.560**	0.538	0.552	**0.745**	**0.780**	tc10	P1	**0.598**	0.550	0.538	0.625	**0.692**
	P2	0.550	**0.615**	**0.598**	0.672	0.742		P2	0.538	0.510	0.535	0.590	0.588
	V	**0.632**	0.518	0.535	**0.688**	**0.700**		V	**0.650**	0.538	0.552	0.702	0.708
tc11	P1	0.628	0.542	0.522	0.618	0.685	tc12	P1	0.572	0.530	0.565	0.708	0.700
	P2	0.630	**0.552**	**0.558**	0.638	0.678		P2	0.625	**0.802**	**0.738**	**0.802**	**0.752**

5.4.3 Discussion

In the introduction of this section, we formulated **RQ3.3**:

> **RQ3.3.** How to use class-based information to efficiently initialize entity and relation embeddings and how profitable is it for KG-based tasks?

In the context of enhancing KGEs for various tasks, leveraging class-based information for initializing entity and relation embeddings emerges as a significant strategy to improve the efficiency and effectiveness of KGE-based tasks. The exploration presented in this section addresses **RQ3.3** by focusing on the utilization of schema-based information, specifically class-based information, to optimize the initialization process of entity and relation embeddings and its impact on the performance of KG-based tasks.

The proposed approach, MASCHInE, leverages schema information in a model-agnostic way, making it applicable across different KGEMs. By generating a protograph from the KG schema that encapsulates general knowledge about entity types, as well as relation domains and ranges, the approach enables a more informed initialization of embeddings. This method not only accelerates the convergence towards optimal model parameters but also enhances the semantic capabilities of the embeddings. Our experiments demonstrate that initializing embeddings with class-based information leads to significant improvements in various tasks, including link prediction, entity clustering, and node classification.

For link prediction, the MASCHInE framework shows an ability to produce embeddings that better capture semantic correctness, as indicated by improved Sem@K values, without sacrificing rank-based metrics performance. In entity clustering and node classification tasks, the initialized embeddings exhibit enhanced capabilities in grouping entities according to their semantic similarities and correctly classifying nodes based on the defined criteria. These findings highlight the versatility of the initialized embeddings and underscore the potential of schema-based information in generating more semantically rich and task-agnostic embeddings.

The work presented in Section 5.4.1 provides a comprehensive examination of the benefits of incorporating class-based information into the initialization of KGEs. By presenting a novel framework and demonstrating its efficacy across multiple KG-based tasks, the research contributes valuable insights into the development of more efficient and semantically aware KGE methods.

5.5 Conclusion and future work

This chapter directly built on the previous one, which introduced Sem@K as an additional metric to provide a more comprehensive evaluation framework for knowledge graph embedding models. The present chapter goes beyond the mere evaluation of the semantic capabilities of such models. More precisely, it is concerned with improving the aforementioned semantic capabilities by designing training approaches leveraging schema-based information. In other words, this chapter proposes to move from schema-agnostic to schema-enhanced models, and to assess the relative gain in performance.

5.5.1 Recap

To this respect, a first contribution was to study type-constrained negative sampling in the specific recommendation case. This use case was insightful as it revealed that injecting schema-based information during training does not necessarily lead to improved semantic awareness (here,

in terms of recommendations). Although this result seems counter-intuitive at first glance, it allows to nurture further analyses and discussions.

In particular, we transitioned to the more general link prediction task, and designed a training approach leveraging information about entity types, relation domains and ranges into the loss functions. This second contribution was not only theoretical – as it first involved tailoring the vanilla loss functions – but also experimental: based on the negative results provided by the recommendation case, we intended to take a different approach on the problem. In this case, we observed a substantial improvement in the semantic capabilities of our knowledge graph embedding models. Importantly, there was also an improvement in terms of rank-based metrics in most cases, which naturally entails us to advocate for neuro-symbolic approaches as a way to not only improve performance in semantic tasks, but also more fundamentally improve embedding-based models for link prediction.

Our last contribution aimed at taking a step further, and examining whether embeddings trained for the link prediction task with the help of a schema could be better used in other tasks such as entity clustering and node classification. To support our research endeavours, class-based information was this time injected at embeddings' initialization. Our lightweight framework named MASCHInE showcased high efficiency. In particular, we demonstrated that it led to better semantic capabilities in link predictions. More importantly, the resulting embeddings proved better at clustering and classification tasks than if they were specifically trained for these tasks in a schema-agnostic way. This result underlines the potential for generating versatile embeddings that can be used in several downstream tasks, with the help of semantically rich resources to help guide and enhance the overall learning process.

5.5.2 Future work

In this chapter, several approaches for developing neuro-symbolic training approaches were discussed. However, our contributions can be further improved in several ways:

Type-constrained negative sampling for recommendation: in-depth analysis. In our experiments, we clearly demonstrated that injecting schema-based information during negative sampling did not necessarily lead to better Sem@K values. In some cases, it can even be worse than the vanilla training procedure which discards any ontological information altogether. In this thesis, we chose to transition to other approaches – mainly studying how schema-based information could fit into other components of the overall training pipeline and what benefits can be expected. In other words, we postponed the analysis of why exactly our first experiment around negative sampling was inconclusive. Therefore, we will aim at conducting more experiments in this direction, and also consider deepening into theoretical considerations to better explain the results we had.

Wider ontological coverage for semantic-driven loss functions. The semantic-driven loss functions we designed proved useful both in terms of Sem@K and rank-based metrics. However, results can be further improved by considering additional ontological properties such as `equivalentClass`, `equivalentProperty`, `inverseOf`, and `subClassOf`. In future work, we will study how the proposed loss functions can accommodate such types of ontological constraints.

Handling incomplete or missing information. It should be pointed out that our datasets have been filtered so that each relation profile is complete, *i.e.* is characterized by both an existing domain and range. In future work, we will adapt our procedure so that

more realistic settings can be considered, especially when relations have a partial profile (*e.g.*, only a domain or a range) or a missing one.

MASCHInE framework extension. In future work, we will extend our proposed MASCHInE framework by adding new protograph design principles, datasets and models. While in the current form, the transfer from the protograph embeddings to the KG entity embeddings is done once, we will study alternatives for their interaction to facilitate iterative co-training. This could potentially lead to a training paradigm better suited for the LP task, which, in our experiments, did not provide results as promising as for the node classification and entity clustering tasks.

Chapter 6

Conclusion and perspectives

Contents

6.1 Summary of contributions

In this thesis, we studied two fundamental areas related to KGs. The first one – the Semantic Web – can be described as the ecosystem that defines and sustains knowledge graphs. This ecosystem is equipped with a set of protocols, representation languages (such as OWL, RDF, RDFS, etc.), and technologies (*e.g.*, SPARQL) that enable the representation, maintenance, and querying of knowledge graphs. The second area we investigated is knowledge graph embedding models. These machine learning-based models learn to encode the best representation possible for each constituent of knowledge graphs, so that these resulting embeddings can be further used in various tasks. One of such tasks is link prediction; it consists in inferring new links to be added to an existing knowledge graph. Notably, this task is commonly addressed using knowledge graph embedding models. More specifically, the main motivation of this thesis is to combine the rich semantics and expressiveness of the Semantic Web with the learning abilities of embedding models. This objective falls within a broader field known as neuro-symbolic AI, which aims to integrate neural network-based methods with human-like, symbolic knowledge-based approaches. By leveraging both types of intelligence, neuro-symbolic AI aims at addressing the weaknesses of each one, setting the path for general approaches that are more aligned with human cognition and thereby more interpretable. In this thesis, we are interested in developing neuro-symbolic approaches for knowledge graph-based applications, focusing on how knowledge graph embeddings align with knowledge derived from schemas.

In Chapter 3, we presented our contributions to enhancing Semantic Web resources, crucial for the advancement of neuro-symbolic AI. We first addressed a significant gap in link prediction datasets, enriching mainstream datasets with necessary schemas and making them publicly available. Our second initiative involved developing EducOnto and EduKG – an ontology and

knowledge graph for education – to aid in university course recommendations for high school students. Finally, we unveiled PyGraft, a versatile tool for creating adaptable synthetic schemas and knowledge graphs, aimed at diversifying benchmark datasets and bolstering neuro-symbolic AI methods in various application domains.

In Chapter 4, we critiqued the sole reliance on rank-based metrics for evaluating link prediction models in knowledge graphs. We aimed at alleviating this issue by proposing Sem@K, a novel semantic-oriented metric. Sem@K is designed to measure a model's ability to identify candidates that align with the domain or range of a relation and it can be adapted to various knowledge graph types. We conducted a comprehensive re-evaluation of popular knowledge graph embedding models using Sem@K, across multiple datasets and models, revealing that rank-based performance does not necessarily correlates with semantic awareness. Furthermore, we illustrated Sem@K's practical application in recommendation systems, such as curriculum and publication venue recommendations, showcasing its effectiveness in filtering items and overcoming the limitations of traditional methods. This positions Sem@K as a vital, complementary metric in the evaluation of knowledge graph embeddings.

In Chapter 5, building on the proposed Sem@K metric, we shifted our focus to enhancing models' semantic capabilities by incorporating schema-based information into their training. We first explored type-constrained negative sampling in recommendations, uncovering that schema information does not always enhance semantic awareness – a finding that prompted further analysis. We then developed a training approach that integrates entity types, relation domains, and ranges into loss functions for general link prediction. This approach led to significant improvements in both semantic capabilities and rank-based metrics. Lastly, we examined the applicability of schema-enhanced embeddings in tasks like entity clustering and node classification. Our introduced MASCHInE framework demonstrated that schema-enriched embeddings outperform schema-agnostic ones in such tasks. This underscores the potential of embeddings trained with semantically rich resources and opens promising research directions around embedding versaility.

6.2 Limitations and open challenges

Each chapter ends with contributions (Chapters 3, 4, and 5) ends with a section dedicated to the limitations and future work that are specific to the presented contributions. In this present section, we aim at providing general limitations that prevail at the level of this dissertation.

While this thesis motivates the use of schema-enriched datasets for link prediction, it should be noted that our contribution to enriching mainstream datasets (Chapter 3) is voluntarily restricted to the following constructs: `rdfs:domain`, `rdfs:range`, `rdf:type`, and `rdfs:subClassOf`. In other words, the augmented datasets we propose contain information about the domain and range of relations, entity types, and the inheritance relations defining class hierarchy. However, not all knowledge graphs contain or even such constructs: for instance, domain and range properties are not directly defined in Wikidata [62]. In addition, several other constructs exist and are not considered by our enrichment procedure, such as logical characteristics of properties (`owl:TransitiveProperty`, `owl:SymmetricProperty`), cardinality constraints (*e.g.* `owl:maxCardinality`, `owl:minCardinality`), and individual identity (*e.g.* `owl:sameAs`, `owl:differentFrom`).

Besides, a major contribution of this thesis lies in the development and presentation of Sem@K to measure the semantic validity of predictions made by a model (Chapter 4). One of our main claim is that the link prediction task deserves a more comprehensive evaluation framework, and that Sem@K perfectly fits into this goal at assessing different aspects of a link

prediction model. However, Sem@K is only concerned with how and to what extent predicted entities comply with relation domains and ranges. While focusing on *semantics*, we should not forget that many other dimensions of embedding-models would ideally need to be assessed. For instance, new metrics could be envisioned for measuring the interpretability of embeddings [143], or the environmental impact of the training and inference procedures of popular embedding-based models [132].

The lastly introduced contributions of this thesis (Chapter 5) address the issue of neuro-symbolic training approaches to improve performance in knowledge graph-based applications. A major limitation which stems from our limited scope of ontological constructs (`rdfs:domain`, `rdfs:range`, `rdf:type`, and `rdfs:subClassOf`) is that the approaches proposed in this dissertation are not evaluated in an ideal environment where a plethora of different constructs would be considered. Consequently, we did not measure the full potential of our proposed learning approaches and it would have been interesting knowing what constructs matter the most, and in which specific scenario. More generally, one could also argue against our experimental procedures, where, in most cases, our novel and neuro-symbolic approaches are benchmarked w.r.t. baselines that do not include any schema-based information. Comparing our contributions to other neuro-symbolic approaches as well as pure symbolic models (*e.g.* rule-based models) is a natural extension of our work. Ultimately, one may wonder why we chose to inject schema-based information during training – while a simple method that filters out semantically invalid candidates (for instance, by assigning them a negatively infinite score) would likely provide similar or better results. However, this thesis seeked to demonstrate that incorporating ontological information at training improve predictive performance. Therefore, the proper way to do so is still to hide this information at inference time and only use it during training and evaluation.

6.3 Perspectives

In a similar manner as what we did in the previous section, we limit ourselves to outlining general perspectives related to the work presented in this dissertation. For more specific and short-term research directions based on the contributions presented in this thesis, we invite the reader to refer to the final section of each chapter.

6.3.1 Towards more realistic settings for link prediction

Transductive link prediction, while valuable, is limited to the scope of pre-existing entities and relations for which embeddings have been learned during training. This limitation is a substantial drawback in dynamic settings where the introduction of new entities and relations is common, rendering transductive methods less effective or even obsolete over time.

In contrast, inductive link prediction is designed to generalize beyond the training set, enabling it to effectively manage unseen data. This approach is crucial in scenarios where it is impractical or impossible to retrain models frequently with updated data. Inductive link prediction holds significant importance as it aligns more closely with real-world scenarios where novel entities and relationships emerge over time. In practical applications, data is dynamic and constantly evolving. For instance, in social networks, new users join and form connections. In public and collaborative (*e.g.* Wikidata) or industrial and proprietary knowledge graphs (*e.g.* NetflixGraph), new concepts, individuals, and relationships are continuously added. As such, inductive link prediction has been gaining ground in recent years due to its practical relevance (Fig. 6.1).

Although numerous approaches have been proposed to tackle the inductive link prediction

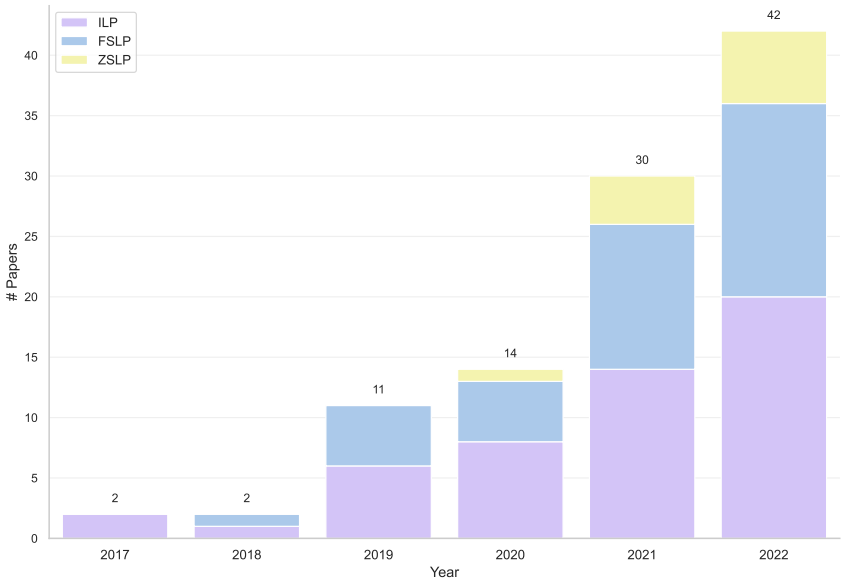


Figure 6.1: Number of papers related to inductive link prediction over the years [73]. *Note*: this figure differentiates inductive link prediction (ILP), from zero-shot (ZSLP) and few-shot link prediction (FSLP). All of them are related and arguably go beyond the transductive setting. For the sake of clarity, we do not introduce any difference between them in the present thesis

task [73], one is of particular interest in the context of this thesis and relies on the potential of leveraging ontologies. For instance, in OntoZSL [51] class and relation representations from the ontological schema are learnt using an ontology embedding method which considers the structural relationships between concepts. Similarly, DOZSL [50] is a model that disentangles the embeddings of ontological concepts in order to obtain finer-grained inter-class relationships between seen and unseen relations. More recently, RMPI [52] also tackles inductive link prediction by leveraging the relation semantics defined in ontological schemas.

Although using semantic information for training embedding models for (transductive) link prediction was not considered a necessity due to the huge amount of relational data available, in scarcer link prediction settings characterized by a paucity of knowledge about some entities and relations, leveraging external sources of information such as schemas and ontologies might become prevalent.

6.3.2 Towards interpretable embeddings

A common tacit assumption is the *entity similarity assumption*, which states that knowledge graph embedding models retain the graph's structure within their embedding space, *i.e.*, position similar entities close to one another. This desirable property make embedding models widely used in downstream tasks such as recommender systems or drug repurposing. In this dissertation, we also extensively use this assumption with the implicit belief that our embedding model will automatically learn how entities relate to one another, and gradually update their respective embeddings to reflect their similar semantics. As human beings, we expect that such automatic approach would somehow generate meaningful clusters of entities, and one could be able to plot the resulting embeddings to easily interpret results. However, up to now, although few works investigate the abilities of embedding models to capture the underlying semantics of entities and

relations, the results presented in [86] and [74] seem to go against this assumption that we tend to take for granted. In particular, Jain *et al.* [86] demonstrated that semantic representation in embeddings is not consistent across all entities in a knowledge graph, but is restricted to a small subset of them. Most importantly, their experimental results cast doubt on the usefulness of using embedding models for performing semantic tasks. In [74], we take a different perspective to explore this tacit assumption: they explore how graph-based similarity correlates with embedding-based similarity, and whether distinct embedding models expose a different notion of similarity. A major finding is that the choice of the model significantly influences the notion of entity similarity encoded in the resulting vector space. This finding has profound implications for a variety of downstream tasks where accurate entity similarity is crucial, *e.g.* recommender systems and semantic searches. In addition, these findings should prompt us to be even more cautious when using embeddings for explainability or interpretability purposes.

6.3.3 On the importance of embeddings

Last section brought a fair share of pessimism around the notion of embeddings and its use in semantic-related tasks. This section, in contrast, seeks to end this thesis on a hopeful and optimistic note, emphasizing the significance of embeddings in both the medium and long-term perspectives.

This thesis makes extensive use of embeddings, to the extent that their presence tends to be overlooked. Similarly, we have focused on embedding-based models for applications on knowledge graphs, at the risk of overshadowing the broader acceptance of the concept of embeddings, which permeates a large part of the literature in machine learning.

In fact, we return to the fundamental notion of embeddings and discuss more generally its use in machine learning. Initially, embeddings were rudimentary. They started as a way to map categorical data (like words, products, or users) into continuous vector spaces. The idea was to transform discrete, often symbolic data into a form where machine learning algorithms could process them more effectively. A major milestone was the development of word embeddings, such as Word2Vec [109] and GloVe [133]. These methods represented words as dense vectors in a continuous space where the distance and direction between vectors captured semantic relationships. Nowadays, embeddings are used in a wide range of applications beyond NLP, like recommendation systems, image recognition, and more.

The advent of large language models (LLMs) such as GPT-3 [23] and its successors represents a significant leap in the artificial intelligence (AI) landscape. These models, trained on vast corpora of text, excel in tasks ranging from text completion to complex question-answering, demonstrating a remarkable grasp of linguistic nuances and world knowledge. However, they are not without limitations. LLMs can struggle with information retrieval accuracy, staying up-to-date with current events, and efficiently handling domain-specific knowledge. This is where the integration of embeddings and innovative architectures like the Retrieval-Augmented Generation (RAG) [101] model comes into play.

RAG combines the generative power of LLMs with the precision of information retrieval, leveraging embeddings to connect the model with external knowledge sources, such as vector databases. In fact, when data is transformed into embeddings, it can be efficiently stored in a vector database, a system designed specifically for managing low-dimensional vector representations. Unlike conventional databases, vector databases are adept at performing operations aligned with the nature of vector spaces, *e.g.* searching for vectors with similar properties, identifying nearest neighbors, and executing clustering algorithms. With these new possibilities, RAG addresses a crucial limitation of LLMs: their reliance on the knowledge contained within their

training data, which becomes gradually outdated as time goes and creates a *knowledge boundary* for which re-training is not desirable.

Another example is the use of dense vector databases for enhancing LLMs' capability in specific domains, allowing the LLM to retrieve and reference highly relevant information efficiently [152].

As generative AI models become more complex, their reliance on efficient data representation, like embeddings, is expected to intensify. This relationship is crucial not only for efficiency but also for enhancing the interpretability of AI, a key challenge in the field. We briefly touched the notion of interpretability while advocating for a broader evaluation framework for knowledge graph embeddings. We also warned against the blind use of embeddings for explainability purposes. This panorama of interconnected challenges, encompassing embeddings, interpretability, and efficiency, reveals a symbiotic relationship. Enhancing embeddings can lead to better data representation and efficiency, which in turn can improve model interpretability. Conversely, striving for more interpretable models can reveal insights that refine our approach to embeddings and data handling. This dynamic interplay signifies a continuous, iterative process where progress in one aspect feeds into and is fed by advancements in others, underscoring the complexity and interdependence of these fundamental AI components among which embeddings play a pivotal role.

Appendix A

Supplementary material for Chapter 4

A.1 Algorithmic procedure

Algorithm 6 Class Generation

Initialize the set of unconnected classes: $U = \{C_1, C_2, ..., C_m\}$
Initialize the set of linked classes: $L = \emptyset$
$C_1 \leftarrow$ U.POP(), L.ADD(C_1)
C_1 `rdfs:subClassOf root`
while `max_depth` is not satisfied **do** ▷ Fulfilling class hierarchy depth
    $C_i \leftarrow$ U.POP(), L.ADD(C_i)
    C_i `rdfs:subClassOf` C_{i-1}
while $U \neq \emptyset$ **do** ▷ Building class hierarchy
    $C_i \leftarrow$ U.POP(), L.ADD(C_i)
    if random$(0, 1) \leq \alpha$ **then** ▷ Adding stochasticity
        Place it randomly
    else
        Place it s.t. `avg_depth` and `inheritance_ratio` are satisfied
while `current_disj` $<$ `disj_ratio` **do** ▷ Adds `owl:disjointWith`
    Pick classes A and B s.t. B is neither a parent nor a child of A
    Make A and B disjoint and extend disjointness to their respective children

Algorithm 7 Relation Generation

Initialize the set of unqualified relations: $U = \{R_1, R_2, ..., R_n\}$
Initialize the set of qualified relations: $Q = \emptyset$
Load compatible patterns
while $U \neq \emptyset$ and attribute proportions not satisfied **do** ▷ Adding attributes
    $R_i \leftarrow$ U.POP(), Q.ADD(R_i)
    QUALIFY(R_i) based on priority order
INVERSE_PAIRING(Q) ▷ Adding `owl:inverseOf`
RELATION_PROFILING(Q) ▷ Adding `rdfs:domain/range`
SUBPROPERTY_PAIRING(Q) ▷ Adding `rdfs:subPropertyOf`

Algorithm 8 Knowledge Graph Generation

Initialize the set of unobserved entities: $U = \{E_1, E_2, ..., E_p\}$
Initialize the set of observed entities: $O = \emptyset$
Initialize the knowledge graph: $\mathcal{KG} = \emptyset$
Load the underpinning schema $\mathcal{S}$
ASSIGN_CLASS(U) ▷ Specific class attribution based on `prop_untyped`
if `multityping` **then**
    COMPLETE_TYPING(U) ▷ Adding specific classes based on `avg_multityping`
while $U \neq \emptyset$ and `num_triples` not satisfied **do**
    $\mathcal{KG} \leftarrow$ GENERATE_TRIPLES($U, O, \mathcal{S}$)
CHECKING_PROCEDURES($\mathcal{KG}$) ▷ Removing foreseeable inconsistencies
REASONING($\mathcal{KG}$) ▷ HermiT reasoner

Appendix B

Supplementary material for Chapter 5

B.1 Hyperparameters

For datasets with no reported optimal hyperparameters, grid-search based on curated hyperpamaters were performed. The full hyperparameter space is provided in Table B.1. Chosen hyperparameters for each pair of KGEM and dataset are provided in Table B.2.

Table B.1: Hyperparameter search space

Hyperparameters	Range
Batch Size	$\{128, 256, 512, 1024, 2048\}$
Embedding Dimension	$\{50, 100, 150, 200\}$
Regularizer Type	$\{\text{None}, L1, L2\}$
Regularizer Weight (λ)	$\{1e^{-1}, 1e^{-2}, 1e^{-3}, 1e^{-4}, 1e^{-5}\}$
Learning Rate (lr)	$\{0.005, 0.003, 0.001, 0.0005, 0.0003, 0.0001\}$

B.2 Results achieved with the best reported hyperparameters

B.3 Evolution of MRR and Sem@K values with respect to the number of epochs

The evolution of MRR and Sem@K values with respect to the number of epochs is presented in Fig. B.1, B.2 , B.3, B.4, B.5, B.6, and B.7. For equity and clarity sakes, we choose to present 2 KGEMs for each family of model (translational models, semantic matching models, CNNs, and GNNs). Regarding semantic matching models, DistMult and ComplEx are chosen, as the evolution of their MRR and Sem@K values is less erratic than SimplE. The evolution of MRR and Sem@K values for SimplE are made available on the GitHub repository of the datasets[33].

[33] `https://github.com/nicolas-hbt/benchmark-sematk`

Table B.2: Chosen hyperparameters for schema-defined and schemaless KGs used in the experiments. $|\mathcal{B}|$, d, lr, and λ denote the batch size, embedding dimension, learning rate, and regularization weight, respectively. We experimentally found that $L2$ regularizer systematically worked the best. We therefore decide not to refer to it in the table. For ConvKB, the number of filters $\mathcal{T}$ was set to 128. Parameters specific to R-GCN and CompGCN were left as defined in the original implementations. The names of some datasets have been shortened

Model	Hyperpar.	FB15k237	DB93k	YAGO3	YAGO4	CoDEx-S	CoDEx-M	WN18RR
TransE	$\|\mathcal{B}\|$	512	256	256	512	128	128	512
	d	100	200	150	100	100	100	100
	lr	$1e^{-3}$	$5e^{-3}$	$5e^{-3}$	$5e^{-2}$	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$
	λ	$1e^{-5}$	$1e^{-5}$	$1e^{-5}$	$1e^{-5}$	$1e^{-5}$	$1e^{-5}$	$1e^{-5}$
TransH	$\|\mathcal{B}\|$	512	256	256	512	128	128	512
	d	100	200	150	100	100	100	100
	lr	$1e^{-3}$	$5e^{-3}$	$5e^{-3}$	$5e^{-2}$	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$
	λ	$1e^{-5}$	$1e^{-5}$	$1e^{-5}$	$1e^{-5}$	$1e^{-5}$	$1e^{-5}$	$1e^{-5}$
DistMult	$\|\mathcal{B}\|$	1024	1024	1024	1024	1024	1024	1024
	d	100	200	150	100	100	100	100
	lr	$1e^{-3}$	$1e^{-3}$	$5e^{-3}$	$5e^{-2}$	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$
	λ	$1e^{-5}$	$1e^{-5}$	$1e^{-5}$	0	0	0	0
ComplEx	$\|\mathcal{B}\|$	1024	1024	1024	1024	1024	1024	1024
	d	100	200	150	100	100	100	100
	lr	$1e^{-3}$	$5e^{-3}$	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$
	λ	$1e^{-2}$	$1e^{-1}$	$1e^{-5}$	$1e^{-2}$	$1e^{-2}$	$1e^{-2}$	$1e^{-2}$
SimplE	$\|\mathcal{B}\|$	1024	1024	1024	1024	1024	1024	1024
	d	100	200	150	100	100	100	100
	lr	$1e^{-3}$	$5e^{-3}$	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$
	λ	$1e^{-2}$	$1e^{-1}$	$1e^{-5}$	0	$1e^{-2}$	$1e^{-2}$	0
ConvE	$\|\mathcal{B}\|$	512	256	256	512	512	512	512
	d	200	200	200	200	200	200	200
	lr	$1e^{-3}$	$5e^{-3}$	$5e^{-3}$	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$
	λ	$1e^{-5}$	$1e^{-5}$	$1e^{-5}$	$1e^{-5}$	$1e^{-5}$	$1e^{-5}$	$1e^{-5}$
ConvKB	$\|\mathcal{B}\|$	512	256	256	512	512	512	512
	d	100	100	100	100	100	100	100
	lr	$1e^{-3}$	$1e^{-3}$	$5e^{-3}$	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$
	λ	$1e^{-5}$	$1e^{-5}$	$1e^{-5}$	$1e^{-5}$	$1e^{-5}$	$1e^{-5}$	$1e^{-5}$
R-GCN	d	500	500	500	500	500	500	500
	lr	$1e^{-2}$	$1e^{-2}$	$1e^{-2}$	$1e^{-2}$	$1e^{-2}$	$1e^{-2}$	$1e^{-2}$
	λ	$1e^{-2}$	$1e^{-2}$	$1e^{-2}$	$1e^{-2}$	$1e^{-2}$	$1e^{-2}$	$1e^{-2}$
CompGCN	$\|\mathcal{B}\|$	1024	1024	1024	1024	1024	1024	1024
	d	200	200	200	200	200	200	200
	lr	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$
	λ	0	0	0	0	0	0	0

Table B.3: Rank-based and semantic-based results achieved at the best epoch in terms of MRR on the schema-defined knowledge graphs. Bold fonts indicate which model performs best with respect to a given metric

(a) FB15k237-ET

		Rank-based				Sem@K[base]			Sem@K[wup]			Sem@K[ext]		
Model Family	Model	MRR	H@1	H@3	H@10	S@1	S@3	S@10	S@1	S@3	S@10	S@1	S@3	S@10
Geometric	TransE	.273	.182	.302	.453	.978	.964	.949	.983	.972	.961	.888	.870	.845
	TransH	.281	.187	.306	.458	.981	.968	.955	.986	.976	.966	.896	.873	.855
Semantic Matching	DistMult	.289	.206	.313	.456	.907	.919	.922	.919	.930	.934	.842	.853	.849
	ComplEx	.267	.185	.292	.431	.874	.823	.761	.897	.856	.804	.822	.770	.7005
	SimplE	.257	.180	.275	.416	.895	.878	.848	.911	.896	.869	.839	.817	.776
Convolutional	ConvE	.310	.211	.337	.490	.972	.970	.961	.977	.973	.968	.883	.880	.870
	ConvKB	.251	.159	.261	.411	.918	.897	.872	.935	.918	.899	.842	.812	.781
	R-GCN	.238	.153	.257	.412	.980	.970	.953	.985	.976	.961	.902	.889	.861
	CompGCN	**.337**	**.227**	**.365**	**.530**	**.998**	**.996**	**.992**	**.999**	**.997**	**.995**	**.993**	**.986**	**.976**

(b) DB93k

		Rank-based				Sem@K[base]			Sem@K[wup]			Sem@K[ext]		
Model Family	Model	MRR	H@1	H@3	H@10	S@1	S@3	S@10	S@1	S@3	S@10	S@1	S@3	S@10
Geometric	TransE	.233	.145	.275	.397	.985	.975	.961	.991	.987	.979	.767	.730	.689
	TransH	.236	.147	.278	.399	**.988**	**.979**	**.968**	**.993**	**.990**	**.981**	.769	.739	.694
Semantic Matching	DistMult	.261	.202	.287	.369	.865	.831	.790	.890	.865	.833	.716	.670	.617
	ComplEx	.287	.213	.325	.417	.941	.917	.877	.955	.937	.907	.815	.767	.698
	SimplE	.252	.202	.274	.339	.896	.871	.837	.915	.895	.867	.774	.719	.659
Convolutional	ConvE	.256	.183	.289	.392	.871	.862	.862	.882	.874	.874	.764	.750	.734
	ConvKB	.178	.121	.199	.283	.883	.895	.881	.902	.895	.881	.742	.702	.659
	R-GCN	.208	.144	.233	.319	.969	.959	.945	.979	.972	.961	.819	.785	.740
	CompGCN	**.319**	**.249**	**.351**	**.446**	**.988**	.976	.967	.991	.982	.975	**.953**	**.918**	**.878**

(c) YAGO3-37k

		Rank-based				Sem@K[base]			Sem@K[wup]			Sem@K[ext]		
Model Family	Model	MRR	H@1	H@3	H@10	S@1	S@3	S@10	S@1	S@3	S@10	S@1	S@3	S@10
Geometric	TransE	.184	.080	.198	.408	.989	.988	.988	.993	.994	.995	.897	.904	.911
	TransH	.187	.091	.199	.415	.995	.993	.993	.995	.997	.997	.901	.911	.925
Semantic Matching	DistMult	.225	.112	.251	.465	.885	.940	.959	.921	.962	.980	.795	.851	.886
	ComplEx	.430	.250	.551	.780	.740	.820	.859	.911	.935	.950	.662	.735	.794
	SimplE	.311	.081	.468	.749	.377	.733	.866	.786	.901	.944	.310	.640	.777
Convolutional	ConvE	**.493**	**.350**	**.578**	**.775**	.933	.923	.910	.977	.977	.975	.893	.879	.871
	ConvKB	.305	.162	.352	.631	.979	.979	.978	.990	.990	.990	.899	.896	.904
	R-GCN	.115	.046	.110	.254	.993	.994	.993	.995	.996	.997	.933	.932	.928
	CompGCN	.399	.269	.464	.663	**.998**	**.997**	**.996**	**.999**	**.999**	**.998**	**.998**	**.994**	**.982**

(d) YAGO4-19k

		Rank-based				Sem@K[base]			Sem@K[wup]			Sem@K[ext]		
Model Family	Model	MRR	H@1	H@3	H@10	S@1	S@3	S@10	S@1	S@3	S@10	S@1	S@3	S@10
Geometric	TransE	.749	.651	.833	.901	.975	.890	.843	.980	.914	.870	.844	.647	.547
	TransH	.752	.656	.829	.898	.979	**.909**	**.868**	**.982**	**.931**	**.891**	.848	.656	.564
Semantic Matching	DistMult	.863	.839	.881	.898	.931	.687	.534	.933	.686	.551	.901	.556	.376
	ComplEx	.890	.881	.897	.903	.923	.627	.479	.923	.635	.496	.909	.524	.349
	SimplE	.808	.757	.848	.883	.891	.646	.500	.894	.654	.516	.840	.530	.367
Convolutional	ConvE	.901	.891	**.908**	.913	.977	.858	.844	.977	.869	.854	.948	.731	.709
	ConvKB	.828	.772	.883	.908	.865	.614	.564	.870	.639	.590	.793	.460	.361
	R-GCN	.893	.885	.896	.903	.969	.832	.727	.970	.841	.741	.930	.670	.522
	CompGCN	**.907**	**.903**	**.908**	**.918**	**.980**	.906	.862	.981	.912	.870	**.968**	**.815**	**.725**

Table B.4: Rank-based and semantic-based results achieved at the best epoch in terms of MRR on the schemaless knowledge graphs. Bold fonts indicate which model performs best with respect to a given metric

(a) CoDEx-S

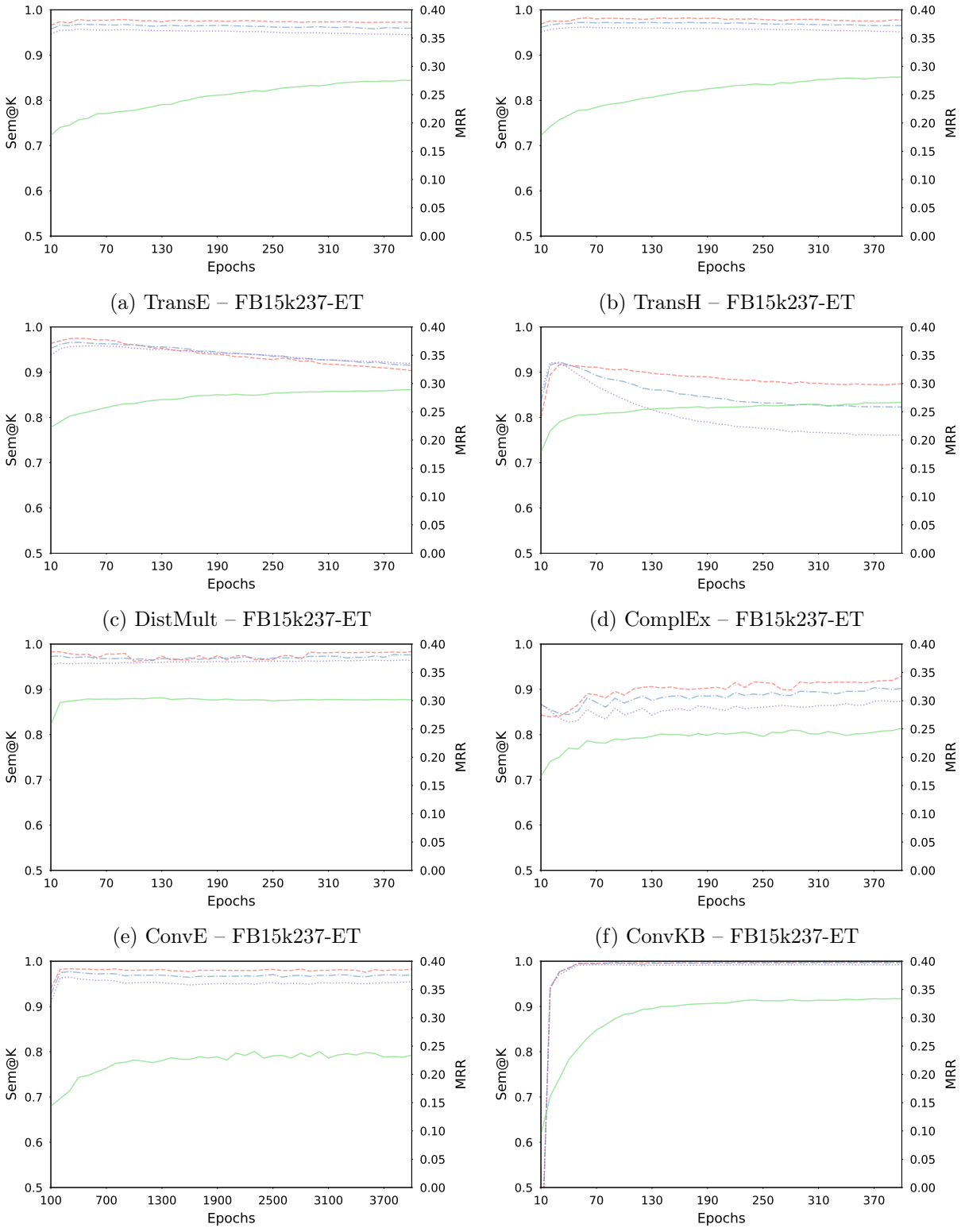
		Rank-based				Sem@K[ext]		
Model Family	Model	MRR	H@1	H@3	H@10	S@1	S@3	S@10
Geometric	TransE	.354	.223	.409	.620	.927	.900	.873
	TransH	.355	.225	.410	.621	.928	.901	.875
Semantic Matching	DistMult	.442	.336	.487	.661	.935	.919	.887
	ComplEx	**.511**	**.426**	**.552**	**.668**	.872	.786	.694
	SimplE	.464	.368	.514	.641	.876	.789	.727
Convolutional	ConvE	.379	.259	.439	.610	.912	.913	.886
	ConvKB	.453	.361	.493	.632	.858	.806	.769
	R-GCN	.335	.225	.381	.556	.936	.921	.889
	CompGCN	.380	.268	.427	.597	**.993**	**.986**	**.974**

(b) CoDEx-M

		Rank-based				Sem@K[ext]		
Model Family	Model	MRR	H@1	H@3	H@10	S@1	S@3	S@10
Geometric	TransE	.262	.188	.284	.405	.889	.855	.839
	TransH	.266	.190	.291	.415	.894	.862	.842
Semantic Matching	DistMult	.268	.199	.291	.402	.726	.749	.765
	ComplEx	.274	.213	.298	.386	.768	.687	.608
	SimplE	.269	.212	.289	.378	.799	.713	.629
Convolutional	ConvE	.255	.184	.280	.395	.863	.839	.814
	ConvKB	.230	.156	.250	.376	.797	.771	.770
	R-GCN	.185	.110	.201	.340	.944	.920	.887
	CompGCN	**.314**	**.237**	**.346**	**.460**	**.996**	**.992**	**.983**

(c) WN18RR

		Rank-based				Sem@K[ext]		
Model Family	Model	MRR	H@1	H@3	H@10	S@1	S@3	S@10
Geometric	TransE	.186	.032	.303	.455	.770	.756	.715
	TransH	.191	.034	.306	.458	.773	.760	.717
Semantic Matching	DistMult	.399	.372	.413	.444	.795	.732	.688
	ComplEx	.430	.400	.446	.487	.715	.669	.673
	SimplE	.397	.375	.406	.438	.690	.598	.673
Convolutional	ConvE	.406	.375	.422	.459	.865	.854	.871
	ConvKB	.356	.302	.391	.444	.712	.713	.732
	R-GCN	.382	.345	.402	.441	.836	.797	.752
	CompGCN	**.471**	**.437**	**.483**	**.536**	**.970**	**.946**	**.927**



(a) TransE – FB15k237-ET

(b) TransH – FB15k237-ET

(c) DistMult – FB15k237-ET

(d) ComplEx – FB15k237-ET

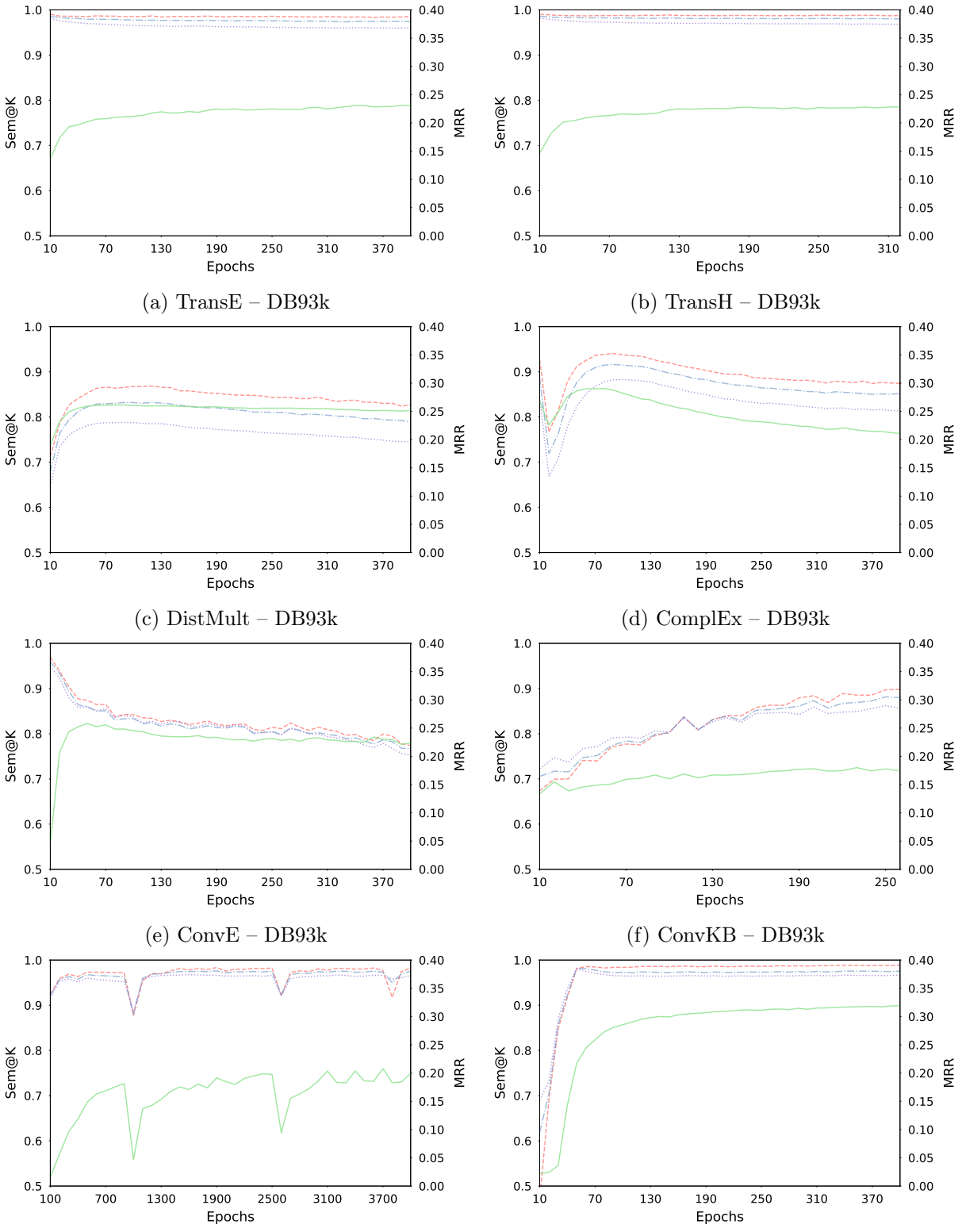
(e) ConvE – FB15k237-ET

(f) ConvKB – FB15k237-ET

(g) R-GCN – FB15k237-ET

(h) CompGCN – FB15k237-ET

Figure B.1: Evolution of MRR (—), Sem@1 (- -), Sem@3 (-·-), and Sem@10 (····) on FB15k237-ET



(a) TransE – DB93k

(b) TransH – DB93k

(c) DistMult – DB93k

(d) ComplEx – DB93k

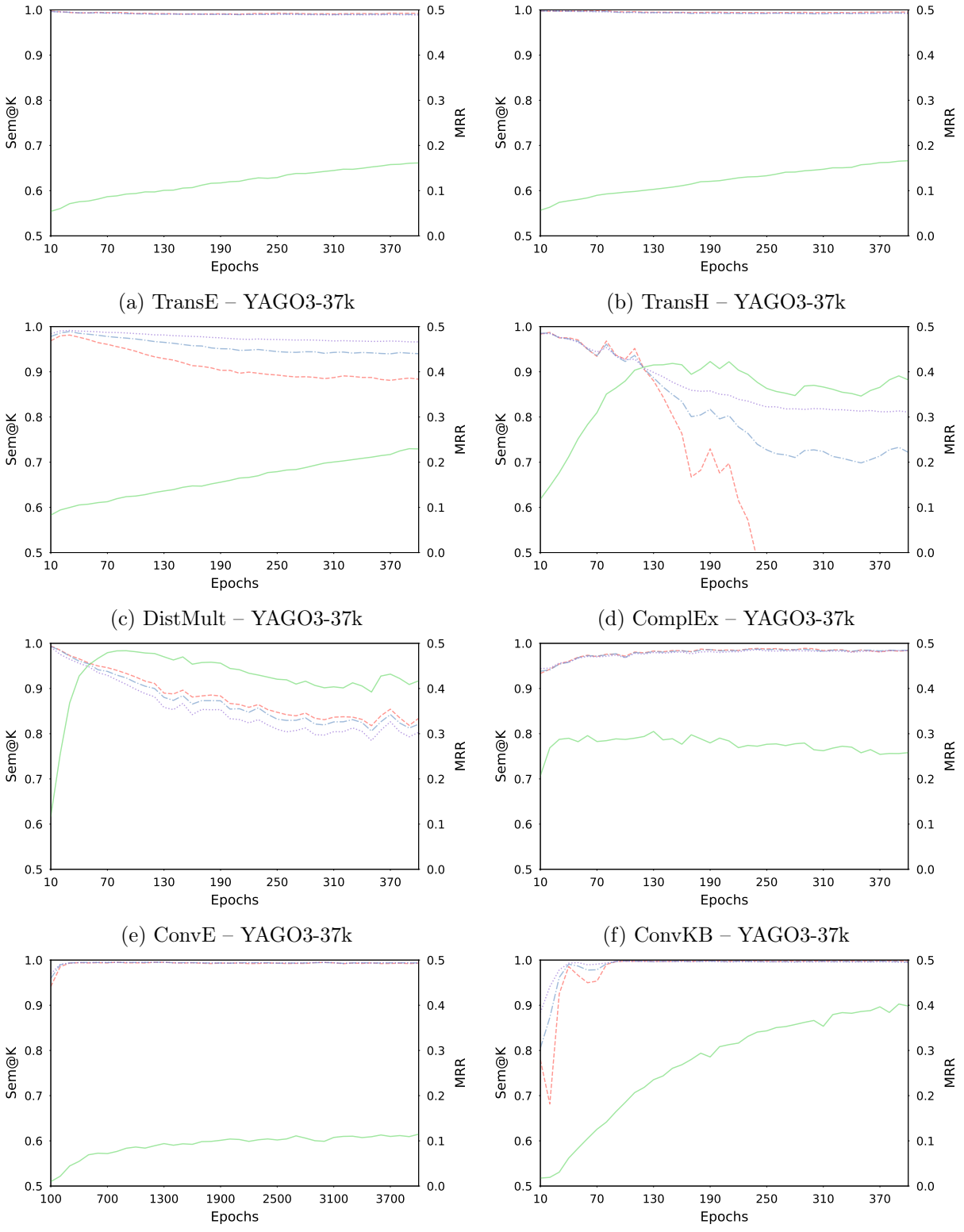
(e) ConvE – DB93k

(f) ConvKB – DB93k

(g) R-GCN – DB93k

(h) CompGCN – DB93k

Figure B.2: Evolution of MRR (—), Sem@1 (- -), Sem@3 (-·-), and Sem@10 (····) on DB93k



(a) TransE – YAGO3-37k

(b) TransH – YAGO3-37k

(c) DistMult – YAGO3-37k

(d) ComplEx – YAGO3-37k

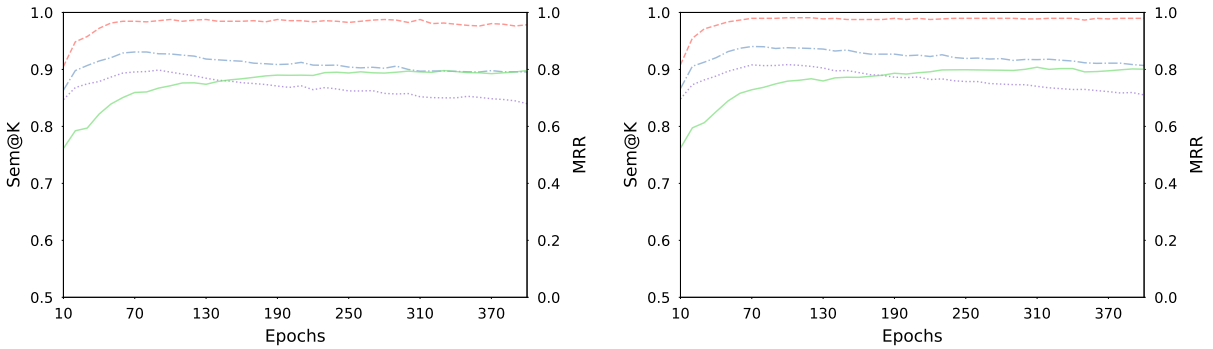
(e) ConvE – YAGO3-37k

(f) ConvKB – YAGO3-37k

(g) R-GCN – YAGO3-37k

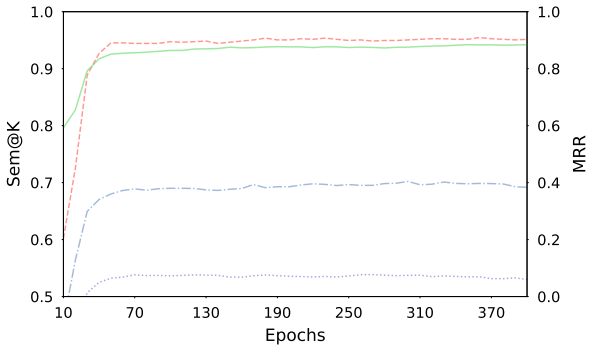
(h) CompGCN – YAGO3-37k

Figure B.3: Evolution of MRR (—), Sem@1 (- -), Sem@3 (-·-), and Sem@10 (····) on YAGO3-37k

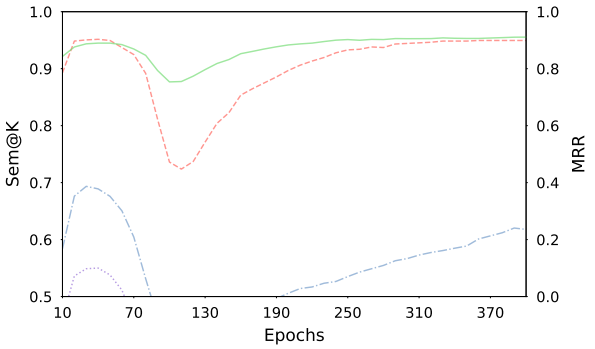


(a) TransE – YAGO4-19k

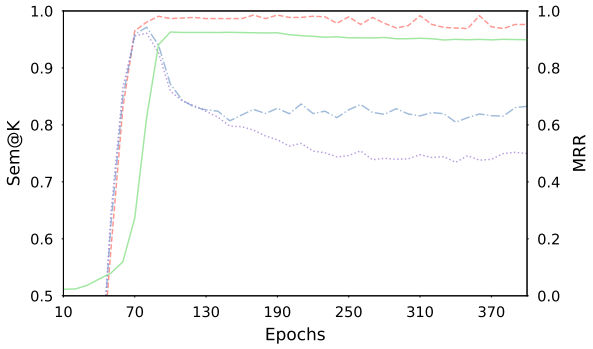
(b) TransH – YAGO4-19k



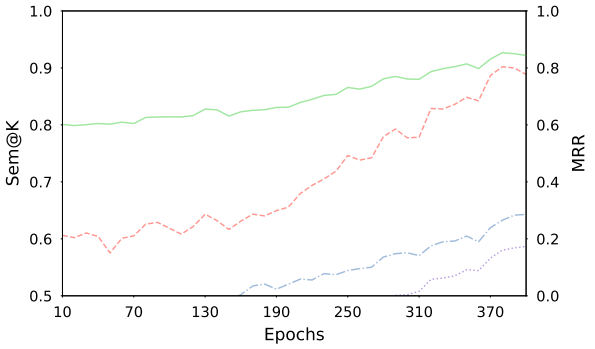
(c) DistMult – YAGO4-19k



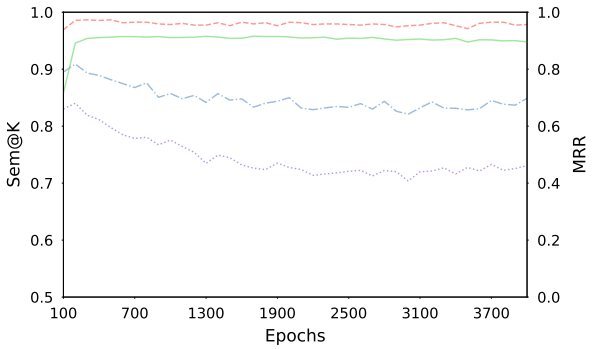
(d) ComplEx – YAGO4-19k



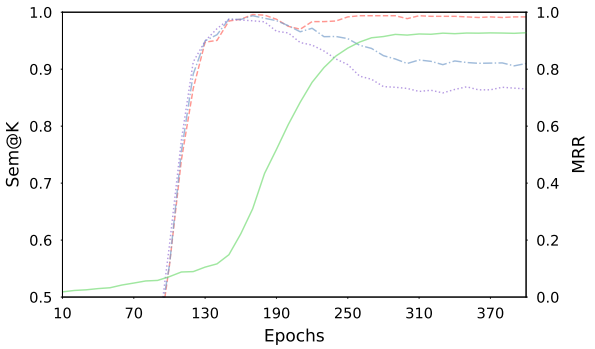
(e) ConvE – YAGO4-19k



(f) ConvKB – YAGO4-19k

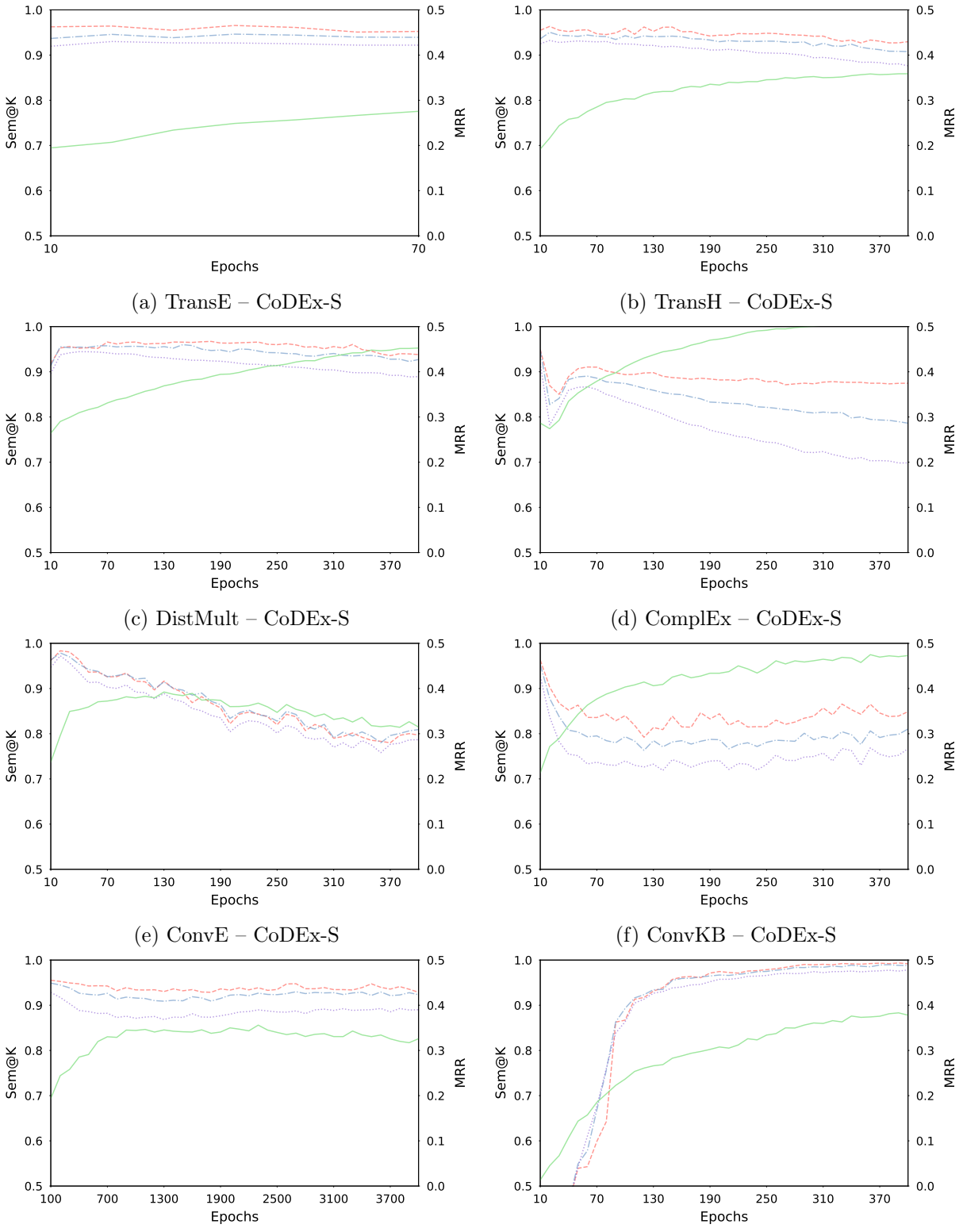


(g) R-GCN – YAGO4-19k



(h) CompGCN – YAGO4-19k

Figure B.4: Evolution of MRR (—), Sem@1 (- -), Sem@3 (-·-), and Sem@10 (····) on YAGO4-19k



(a) TransE – CoDEx-S

(b) TransH – CoDEx-S

(c) DistMult – CoDEx-S

(d) ComplEx – CoDEx-S

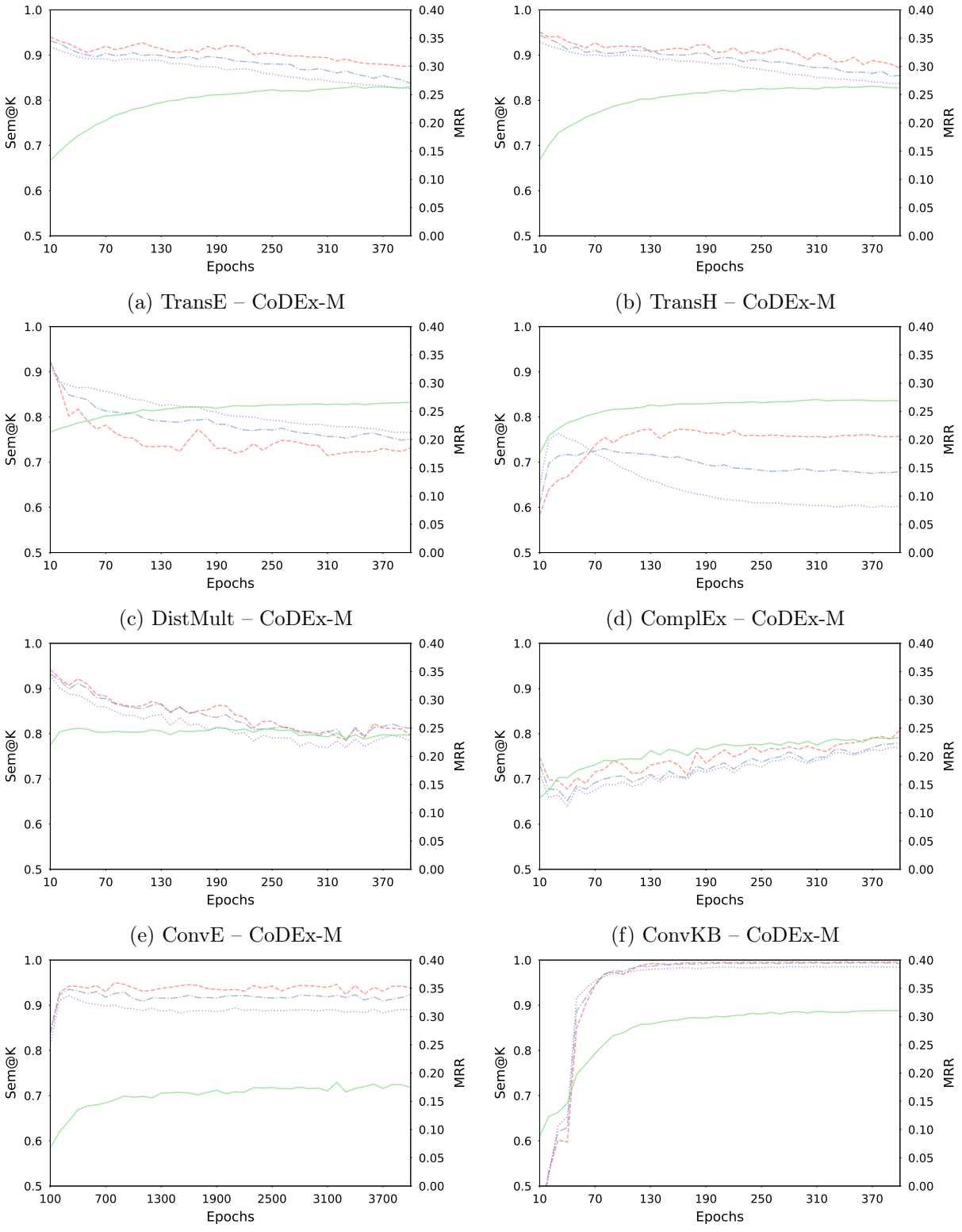
(e) ConvE – CoDEx-S

(f) ConvKB – CoDEx-S

(g) R-GCN – CoDEx-S

(h) CompGCN – CoDEx-S

Figure B.5: Evolution of MRR (—), Sem@1 (- -), Sem@3 (-·-), and Sem@10 (····) on CoDEx-S



(a) TransE – CoDEx-M

(b) TransH – CoDEx-M

(c) DistMult – CoDEx-M

(d) ComplEx – CoDEx-M

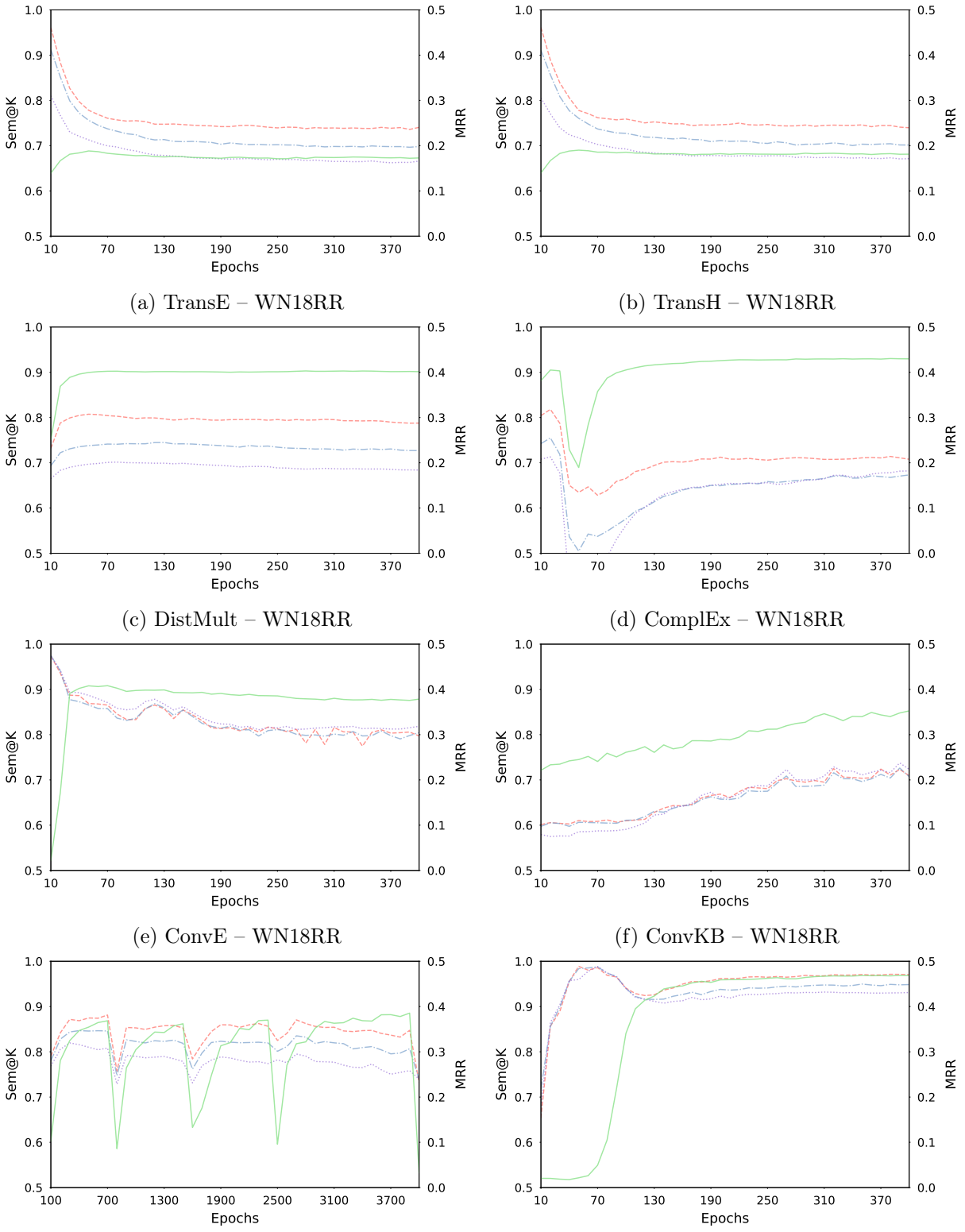
(e) ConvE – CoDEx-M

(f) ConvKB – CoDEx-M

(g) R-GCN – CoDEx-M

(h) CompGCN – CoDEx-M

Figure B.6: Evolution of MRR (—), Sem@1 (- -), Sem@3 (-·-), and Sem@10 (····) on CoDEx-M



(a) TransE – WN18RR

(b) TransH – WN18RR

(c) DistMult – WN18RR

(d) ComplEx – WN18RR

(e) ConvE – WN18RR

(f) ConvKB – WN18RR

(g) R-GCN – WN18RR

(h) CompGCN – WN18RR

Figure B.7: Evolution of MRR (—), Sem@1 (- -), Sem@3 (-·-), and Sem@10 (····) on WN18RR

Appendix C

Supplementary material for Chapter 6

C.1 Hyperparameters

Full hyperparameter search space is reported in Table C.1. Those hyperparameters leading to the best MRR results under the original loss formulation for each KGEM were obtained with grid-search over a defined hyperparameter space. These optimal hyperparameters are reported in Table C.2. Likewise, grid-search was performed for finding the best hyperparameters for $\mathcal{L}^{S'}_{BCEL}$ and $\mathcal{L}^{S'}_{PLL}$, which are reported in Table C.3.

Table C.1: Hyperparameter search space

Hyperparameters	Range
Batch Size	$\{128, 256, 512, 1024, 2048\}$
Embedding Dimension	$\{50, 100, 150, 200\}$
Regularizer Type	{None, $L1$, $L2$}
Regularizer Weight (λ)	$\{1e^{-2}, 1e^{-3}, 1e^{-4}, 1e^{-5}\}$
Learning Rate (lr)	$\{1e^{-2}, 5e^{-3}, 1e^{-3}, 5e^{-4}, 1e^{-4}\}$
Margin γ ($\mathcal{L}_{PHL}$)	$\{1, 2, 3, 5, 10, 20\}$
Semantic Factor ϵ ($\mathcal{L}^{S}_{PHL}$)	$\{0.01, 0.1, 0.25, 0.5, 0.75\}$
Semantic Factor ϵ ($\mathcal{L}^{S}_{PLL}$)	$\{0.05, 0.10, 0.15, 0.25\}$
Semantic Factor ϵ ($\mathcal{L}^{S}_{BCEL}$)	$\{1e^{-1}, 1e^{-2}, 1e^{-3}, 1e^{-4}, 1e^{-5}\}$

C.2 Modified Versions of $\mathcal{L}^{S}_{BCEL}$ and $\mathcal{L}^{S}_{PLL}$

In Table C.4, we report results achieved with KGEMs trained under $\mathcal{L}^{S'}_{BCEL}$ and $\mathcal{L}^{S'}_{PLL}$, where the superscript S' denotes a different way to include semantic information. In fact, $\mathcal{L}^{S'}_{PLL}$ makes use of the modified labelling function as used in $\mathcal{L}^{S}_{BCEL}$ and defined in Equation (5) of the paper. Conversely, $\mathcal{L}^{S'}_{BCEL}$ uses the binary (unmodified) labelling function ℓ but adopts the same procedure as $\mathcal{L}^{S}_{PLL}$: semantically valid negative triples are considered as positive with probability ϵ %. Hyperparameters for $\mathcal{L}^{S}_{BCEL}$ and $\mathcal{L}^{S}_{PLL}$ are reported in Table C.2, while hyperparameters for $\mathcal{L}^{S'}_{BCEL}$ and $\mathcal{L}^{S'}_{PLL}$ are reported in Table C.3. Results achieved with all the aforementioned loss functions are provided in Table C.4. It shows that the semantic-driven loss functions presented in the paper are the most performing ones.

Table C.2: Chosen hyperparameters for the KGEMs as trained with their original loss function, both in vanilla and semantic-driven versions. $|\mathcal{B}|$, d, lr, λ, and ϵ denote the batch size, embedding dimension, learning rate, regularization weight, and semantic factor, respectively. We experimentally found that $L2$ regularizer systematically worked the best. We therefore decide not to refer to it in the table

Model	Hyperparameters	DB77k	FB15k187	YAGO4-14k
TransE	$\|\mathcal{B}\|$	2048	2048	1024
	d	200	200	100
	lr	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$
	λ	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$
	ϵ	$5e^{-1}$	0.25	0.25
TransH	$\|\mathcal{B}\|$	2048	2048	1024
	d	200	200	100
	lr	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$
	λ	$1e^{-5}$	$1e^{-5}$	$1e^{-5}$
	ϵ	$5e^{-1}$	0.25	0.25
DistMult	$\|\mathcal{B}\|$	2048	2048	1024
	d	200	200	100
	lr	$1e^{-1}$	$1e^{1}$	$1e^{-4}$
	λ	$1e^{-5}$	$1e^{-5}$	$1e^{-5}$
	ϵ	$5e^{-1}$	0.25	0.25
ComplEx	$\|\mathcal{B}\|$	2048	2048	1024
	d	200	200	100
	lr	$1e^{-3}$	$1e^{-3}$	$1e^{-2}$
	λ	$1e^{-1}$	$1e^{-1}$	$1e^{-1}$
	ϵ	0.15	0.15	.015
SimplE	$\|\mathcal{B}\|$	2048	2048	1024
	d	200	200	100
	lr	$1e^{-1}$	$1e^{-1}$	$1e^{-1}$
	λ	$1e^{-2}$	$1e^{-1}$	$1e^{-5}$
	ϵ	.15	.15	.15
ConvE	$\|\mathcal{B}\|$	512	128	512
	d	200	200	200
	lr	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$
	λ	0	0	0
	ϵ	$1e^{-4}$	$1e^{-3}$	$1e^{-3}$
TuckER	$\|\mathcal{B}\|$	128	128	128
	d	200	200	100
	lr	$1e^{-3}$	$5e^{-4}$	$1e^{-3}$
	λ	0	0	0
	ϵ	$1e^{-5}$	$1e^{-4}$	$1e^{-4}$
RGCN	d	500	500	500
	lr	$1e^{-2}$	$1e^{-2}$	$1e^{-2}$
	λ	$1e^{-2}$	$1e^{-2}$	$1e^{-2}$
	ϵ	$1e^{-1}$	$1e^{-1}$	$1e^{-1}$

Table C.3: Chosen hyperparameters for the KGEMs trained with the alternative semantic-driven loss function as detailed in Section C.2. $|\mathcal{B}|$, d, lr, λ, and ϵ denote the batch size, embedding dimension, learning rate, regularization weight, and semantic factor, respectively. We experimentally found that $L2$ regularizer systematically worked the best. We therefore decide not to refer to it in the table

Model	Hyperparameters	DB77k	FB15k187	YAGO4-14k
ComplEx	$\|\mathcal{B}\|$	2048	2048	1024
	d	200	200	100
	lr	$1e^{-4}$	$1e^{-4}$	$1e^{-3}$
	λ	$1e^{-1}$	$1e^{-1}$	$1e^{-1}$
	ϵ	$-1e^{-1}$	$-1e^{-1}$	$1e^{-2}$
SimplE	$\|\mathcal{B}\|$	2048	2048	1024
	d	200	200	100
	lr	$1e^{-3}$	$1e^{-4}$	$1e^{-3}$
	λ	$1e^{-1}$	$1e^{-1}$	$1e^{-1}$
	ϵ	$-1e^{-1}$	$1e^{-2}$	$1e^{-2}$
ConvE	$\|\mathcal{B}\|$	512	128	512
	d	200	200	200
	lr	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$
	λ	0	0	0
	ϵ	$1e^{-6}$	$1e^{-5}$	$1e^{-4}$
TuckER	$\|\mathcal{B}\|$	128	128	128
	d	200	200	100
	lr	$1e^{-3}$	$5e^{-4}$	$1e^{-3}$
	λ	0	0	0
	ϵ	$1e^{-6}$	$1e^{-5}$	$1e^{-5}$
RGCN	d	500	500	500
	lr	$1e^{-2}$	$1e^{-2}$	$1e^{-2}$
	λ	$1e^{-2}$	$1e^{-2}$	$1e^{-2}$
	ϵ	$1e^{-4}$	$1e^{-5}$	$1e^{-4}$

C.3 Bucket Analysis

Relations are separated into three non-intersecting buckets : relations that feature narrow (B1), intermediate (B2), and large (B3) sets of semantically valid heads or tails, respectively. Cut-offs are manually defined for placing a given relation in its corresponding bucket. Such buckets are reported in Table C.5. Results achieved on B1 are reported in the paper, while results for buckets B2 and B3 are shown for DB77k, FB15k187, and YAGO4-14k in Table C.6, Table C.7, and Table C.8, respectively.

Table C.4: Rank-based and semantic-based results on FB15k187, DB77k, and YAGO4-14k. Bold fonts indicate which model performs best w.r.t. a given metric. Suffixes S and S' indicate whether the model is trained under the best (as presented in the paper) or the worst (as presented here) semantic-driven version of the loss function, respectively

	FB15k187			DB77k			YAGO4-14k		
	MRR	H@10	S@10	MRR	H@10	S@10	MRR	H@10	S@10
ComplEx-S	**.316**	**.476**	**.796**	**.297**	**.409**	**.897**	**.923**	**.931**	**.667**
ComplEx-S'	.227	.384	.777	.252	.350	.918	.907	.930	.603
SimplE-S	**.268**	**.409**	.759	.230	**.302**	**.850**	**.924**	**.927**	**.769**
SimplE-S'	.169	.288	**.827**	.230	.297	.583	.885	.915	.290
ConvE-S	**.283**	**.476**	**.996**	**.283**	**.405**	**.985**	.933	.940	**.997**
ConvE-S'	.271	.472	.975	.273	.383	.935	.933	**.941**	.894
TuckER-S	**.320**	**.522**	**.996**	**.312**	**.421**	**.969**	**.931**	**.943**	**.929**
TuckER-S'	.316	.517	.983	.311	.412	.912	.918	.938	.867
RGCN-S	**.260**	**.415**	**.860**	**.197**	**.320**	**.957**	**.927**	**.934**	**.828**
RGCN-S'	.243	.391	.780	.146	.246	.862	.912	.922	.385

Table C.5: Cut-offs for FB15k187, DB77k, and YAGO4-14k. B1, B2, and B3 denote the buckets of relations with narrow, intermediate, and large sets of semantically valid heads or tails, respectively. $|\mathcal{R}|$ denotes the number of unique relations in a given bucket and |Sem. Val| indicates the interval of the number of semantically valid entities for the bucket relations. To illustrate, |Sem. Val| = [11, 216] for the head side means that relations in the bucket have at least 11 and at most 216 semantically valid heads

Bucket	Side	FB15k187		DB77k		YAGO4-14k	
		|Sem. Val|	$\|\mathcal{R}\|$	|Sem. Val|	$\|\mathcal{R}\|$	|Sem. Val|	$\|\mathcal{R}\|$
B1	Head	$[11, 216]$	69	$[12, 930]$	62	$[93, 811]$	10
	Tail	$[12, 244]$	80	$[19, 801]$	44	$[35, 678]$	13
B2	Head	$[278, 1391]$	55	$[1295, 11586]$	58	$[2102, 3624]$	15
	Tail	$[278, 1391]$	49	$[1419, 11586]$	55	$[2102, 3624]$	16
B3	Head	$[1473, 4500]$	63	$[22252, 57242]$	25	$\{5730\}$	12
	Tail	$[1473, 4500]$	58	$\{57242\}$	50	$\{5730\}$	8

Table C.6: Rank-based and semantic-based results on DB77k for buckets of relations that feature an intermediate (B2) and large (B3) set of semantically valid heads or tails. Bold fonts indicate which model performs best w.r.t. a given metric. Suffixes V and S indicate whether the model is trained under the vanilla or semantic-driven version of the loss function, respectively. Hits@10 and Sem@10 are abbreviated to H@10 and S@10

	B2			B3		
	MRR	H@10	S@10	MRR	H@10	S@10
TransE-V	**.450**	**.607**	.838	**.317**	**.429**	.995
TransE-S	.404	.556	**.987**	.300	.407	**1**
TransH-V	**.449**	**.610**	.729	**.311**	**.425**	.971
TransH-S	.423	.592	**.981**	.296	.413	**1**
DistMult-V	.446	.553	.669	.505	.413	.742
DistMult-S	**.450**	**.566**	**.790**	**.506**	**.422**	**.920**
ComplEx-V	.442	.538	.551	**.582**	**.453**	.787
ComplEx-S	**.448**	**.545**	**.707**	.505	.426	**.975**
SimplE-V	**.381**	**.461**	**.716**	**.485**	**.357**	.954
SimplE-S	.350	.404	.649	.386	.276	**.960**
ConvE-V	.388	.535	.890	**.489**	.371	.960
ConvE-S	**.429**	**.559**	**.977**	.450	**.399**	**.999**
TuckER-V	.438	.547	.874	**.591**	.436	.898
TuckER-S	**.444**	**.568**	**.923**	.564	**.444**	**.983**
RGCN-V	**.282**	.413	.670	**.367**	.322	.971
RGCN-S	.275	**.423**	**.861**	.362	**.357**	**.999**

Table C.7: Rank-based and semantic-based results on FB15k187 for the buckets of relations that feature an intermediate (B2) and large (B3) set of semantically valid heads or tails. Bold fonts indicate which model performs best w.r.t. a given metric. Suffixes V and S indicate whether the model is trained under the vanilla or semantic-driven version of the loss function, respectively. Hits@10 and Sem@10 are abbreviated to H@10 and S@10

	B2			B3		
	MRR	H@10	S@10	MRR	H@10	S@10
TransE-V	.330	.526	.934	.141	.255	.953
TransE-S	**.385**	**.588**	**.972**	**.169**	**.290**	**.993**
TransH-V	.330	.517	.846	.161	.262	.963
TransH-S	**.380**	**.590**	**.967**	**.171**	**.291**	**.993**
DistMult-V	.336	.527	.780	.177	.274	.946
DistMult-S	**.388**	**.579**	**.962**	**.187**	**.309**	**.995**
ComplEx-V	.327	.476	.318	**.197**	.306	.717
ComplEx-S	**.351**	**.537**	**.769**	.191	**.310**	**.942**
SimplE-V	.283	.432	.331	**.179**	**.274**	.694
SimplE-S	.283	**.448**	**.671**	.159	.243	**.923**
ConvE-V	.347	.529	.974	.172	.277	.977
ConvE-S	**.354**	**.543**	**.998**	**.188**	**.283**	**.999**
TuckER-V	.387	.574	.987	.215	.330	.994
TuckER-S	**.396**	**.585**	**.991**	**.222**	**.337**	**.997**
RGCN-V	.294	.448	.749	.140	.222	.954
RGCN-S	**.303**	**.474**	**.869**	**.157**	**.229**	**.974**

Table C.8: Rank-based and semantic-based results on YAGO4-14k for the buckets of relations that feature an intermediate (B2) and large (B3) set of semantically valid heads or tails. Bold fonts indicate which model performs best w.r.t. a given metric. Suffixes V and S indicate whether the model is trained under the vanilla or semantic-driven version of the loss function, respectively. Hits@10 and Sem@10 are abbreviated to H@10 and S@10

	B2			B3		
	MRR	H@10	S@10	MRR	H@10	S@10
TransE-V	**.879**	**.928**	.892	.841	**.923**	.974
TransE-S	.861	.922	**.997**	**.854**	.917	**1**
TransH-V	.854	**.922**	.567	**.788**	.92	.803
TransH-S	**.865**	.921	**.876**	.778	**.926**	**.996**
DistMult-V	.852	**.915**	**.443**	.941	.911	.536
DistMult-S	**.862**	.911	.441	.941	.911	**.584**
ComplEx-V	**.883**	**.921**	.352	**.932**	**.914**	.619
ComplEx-S	.881	.918	**.738**	.922	.914	**.964**
SimplE-V	.882	.915	.378	**.932**	**.914**	.656
SimplE-S	**.883**	**.918**	**.841**	.930	.905	**.991**
ConvE-V	**.893**	**.928**	.858	**.941**	.917	.904
ConvE-S	.892	.925	**.931**	.939	**.923**	**.956**
TuckER-V	.884	.928	.791	.941	.917	.915
TuckER-S	**.894**	**.935**	**.930**	**.942**	.917	**.983**
RGCN-V	.880	.894	.380	.930	.898	.670
RGCN-S	**.892**	**.925**	**.953**	**.942**	**.923**	**.996**

References

[1] Kian Ahrabian et al. "Structure Aware Negative Sampling in Knowledge Graphs". In: *Proc. of the 2020 Conf. on Empirical Methods in Natural Language Processing, EMNLP*. Association for Computational Linguistics, 2020, pp. 6093–6101.

[2] Farahnaz Akrami et al. "Realistic Re-Evaluation of Knowledge Graph Completion Methods: An Experimental Study". In: *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*. SIGMOD '20. Portland, OR, USA: Association for Computing Machinery, 2020, pp. 1995–2010. ISBN: 9781450367356. DOI: 10.1145/3318464.3380599.

[3] Mehdi Ali et al. "Bringing Light Into the Dark: A Large-Scale Evaluation of Knowledge Graph Embedding Models Under a Unified Framework". In: vol. 44. 12. 2022, pp. 8825–8845. DOI: 10.1109/TPAMI.2021.3124805.

[4] Mehdi Ali et al. "PyKEEN 1.0: A Python Library for Training and Evaluating Knowledge Graph Embeddings". In: vol. 22. 2021, 82:1–82:6.

[5] Sören Auer et al. "DBpedia: A Nucleus for a Web of Open Data". In: *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC + ASWC*. Vol. 4825. Lecture Notes in Computer Science. Springer, 2007, pp. 722–735.

[6] Samy Badreddine et al. "Logic Tensor Networks". In: vol. 303. 2022, p. 103649. DOI: 10.1016/j.artint.2021.103649.

[7] Ivana Balazevic, Carl Allen, and Timothy M. Hospedales. "TuckER: Tensor Factorization for Knowledge Graph Completion". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*. Association for Computational Linguistics, 2019, pp. 5184–5193. DOI: 10.18653/v1/D19-1522.

[8] Luigi Bellomarini et al. "Knowledge Graphs and Enterprise AI: The Promise of an Enabling Technology". In: *35th IEEE International Conference on Data Engineering, ICDE 2019, Macao, China, April 8-11, 2019*. IEEE, 2019, pp. 26–37. DOI: 10.1109/ICDE.2019.00011.

[9] Michael K. Bergman. "A Common Sense of Knowledge Graphs". In: 2019. URL: https://api.semanticscholar.org/CorpusID:204957313.

[10] Max Berrendorf et al. "On the ambiguity of rank-based evaluation of entity alignment or link prediction methods". In: 2020.

[11] Peru Bhardwaj et al. "Poisoning Knowledge Graph Embeddings via Relation Inference Patterns". In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*. Association for Computational Linguistics, 2021, pp. 1875–1888.

[12] Christian Bizer et al. "DBpedia - A crystallization point for the Web of Data". In: vol. 7. 3. 2009, pp. 154–165. DOI: 10.1016/J.WEBSEM.2009.07.002.

[13] Stephan Bloehdorn and York Sure. "Kernel Methods for Mining Instance Data in Ontologies". In: *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007*. Vol. 4825. Lecture Notes in Computer Science. Springer, 2007, pp. 58–71. DOI: 10.1007/978-3-540-76298-0_5.

[14] J. Bobadilla et al. "Recommender systems survey". In: *Knowledge-Based Systems* 46 (2013), pp. 109–132. ISSN: 0950-7051. DOI: https://doi.org/10.1016/j.knosys.2013.03.012.

[15] Kurt Bollacker et al. "Freebase: A Collaboratively Created Graph Database for Structuring Human Knowledge". In: *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*. SIGMOD '08. Vancouver, Canada: Association for Computing Machinery, 2008, pp. 1247–1250. ISBN: 9781605581026. DOI: 10.1145/1376616.1376746. URL: https://doi.org/10.1145/1376616.1376746.

[16] Kurt D. Bollacker et al. "Freebase: a collaboratively created graph database for structuring human knowledge". In: *Proc. of the ACM SIGMOD International Conference on Management of Data*. ACM, 2008, pp. 1247–1250.

[17] Piero Andrea Bonatti et al. "Knowledge Graphs: New Directions for Knowledge Representation on the Semantic Web (Dagstuhl Seminar 18371)". In: vol. 8. 9. 2018, pp. 29–111. DOI: 10.4230/DAGREP.8.9.29.

[18] Antoine Bordes et al. "Translating Embeddings for Modeling Multi-relational Data". In: *Conference on Neural Information Processing Systems (NeurIPS)*. 2013, pp. 2787–2795.

[19] Antoine Bordes et al. "Translating Embeddings for Modeling Multi-relational Data". In: *Conf. on Neural Information Processing Systems (NeurIPS) 2013*. 2013, pp. 2787–2795.

[20] Armand Boschin. "TorchKGE: Knowledge Graph Embedding in Python and PyTorch". In: *International Workshop on Knowledge Graph: Mining Knowledge Graph for Deep Insights*. Aug. 2020.

[21] Ronald J. Brachman and James G. Schmolze. "An Overview of the KL-ONE Knowledge Representation System". In: vol. 9. 2. 1985, pp. 171–216. DOI: 10.1207/S15516709COG0902_1.

[22] Samuel Broscheit et al. "LibKGE - A knowledge graph embedding library for reproducible research". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online: Association for Computational Linguistics, Oct. 2020, pp. 165–174. DOI: 10.18653/v1/2020.emnlp-demos.22.

[23] Tom B. Brown et al. "Language Models are Few-Shot Learners". In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. 2020.

[24] Bruce G. Buchanan and Edward A. Feigenbaum. "Dendral and Meta-Dendral: Their Applications Dimension". In: vol. 11. 1-2. 1978, pp. 5–24. DOI: 10.1016/0004-3702(78)90010-3.

[25] Maritza Bustos López et al. "EduRecomSys: An Educational Resource Recommender System Based on Collaborative Filtering and Emotion Detection". In: *Interacting with Computers* 32.4 (Feb. 2021), pp. 407–432. ISSN: 1873-7951. DOI: 10.1093/iwc/iwab001. eprint: https://academic.oup.com/iwc/article-pdf/32/4/407/38855601/iwab001.pdf.

[26] Borui Cai et al. "Temporal Knowledge Graph Completion: A Survey". In: *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization, Aug. 2023. DOI: 10.24963/ijcai.2023/734.

[27] Liwei Cai and William Yang Wang. "KBGAN: Adversarial Learning for Knowledge Graph Embeddings". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*. Association for Computational Linguistics, 2018, pp. 1470–1480. DOI: 10.18653/v1/n18-1133.

[28] Jiahang Cao et al. "Knowledge Graph Embedding: A Survey from the Perspective of Representation Spaces". In: 2022. DOI: 10.48550/ARXIV.2211.03536.

[29] Zongsheng Cao et al. "ER: Equivariance Regularizer for Knowledge Graph Completion". In: *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*. AAAI Press, 2022, pp. 5512–5520.

[30] Penghe Chen et al. "An Automatic Knowledge Graph Construction System for K-12 Education". In: *Proceedings of the Fifth Annual ACM Conference on Learning at Scale*. New York, NY, USA: Association for Computing Machinery, 2018. ISBN: 9781450358866. DOI: 10.1145/3231644.3231698.

[31] Yao Chen et al. "MöbiusE: Knowledge Graph Embedding on Möbius Ring". In: vol. 227. C. Elsevier Science Publishers B. V., Sept. 2021. DOI: 10.1016/j.knosys.2021.107181.

[32] Zhe Chen et al. "Knowledge Graph Completion: A Review". In: vol. 8. 2020, pp. 192435–192456. DOI: 10.1109/ACCESS.2020.3030076.

[33] Boya Cheng, Yuan Zhang, and Dongxin Shi. "Ontology-Based Personalized Learning Path Recommendation for Course Learning". In: 2018, pp. 531–535.

[34] Luca Costabello et al. *AmpliGraph: a Library for Representation Learning on Knowledge Graphs*. Mar. 2019. DOI: 10.5281/zenodo.2595043.

[35] Claudia d'Amato, Nicola Flavio Quatraro, and Nicola Fanizzi. "Injecting Background Knowledge into Embedding Models for Predictive Tasks on Knowledge Graphs". In: *The Semantic Web - 18th International Conference, ESWC 2021, Virtual Event, June 6-10, 2021, Proceedings*. Vol. 12731. Lecture Notes in Computer Science. Springer, 2021, pp. 441–457. DOI: 10.1007/978-3-030-77385-4_26.

[36] Luc De Raedt et al. "A Hybrid Approach to Learning and Its Knowledge Representation". In: *Proceedings of the Third COGNITIVA Symposium on At the Crossroads of Artificial Intelligence, Cognitive Science, and Neuroscience*. NLD: North-Holland Publishing Co., 1991, pp. 109–116. ISBN: 0444890491.

[37] Klaas Dellschaft and Steffen Staab. "Strategies for the Evaluation of Ontology Learning". In: *Proceedings of the 2008 Conference on Ontology Learning and Population: Bridging the Gap between Text and Knowledge*. NLD: IOS Press, 2008, pp. 253–272. ISBN: 9781586038182.

[38] Gianluca Demartini et al. "The Bowlogna ontology: Fostering open curricula and agile knowledge bases for Europe's higher education landscape". In: vol. 4. Jan. 2013, pp. 53–63. DOI: 10.3233/SW-2012-0064.

[39] Tim Dettmers et al. "Convolutional 2D Knowledge Graph Embeddings". In: *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*. AAAI Press, 2018, pp. 1811–1818.

[40] Boyang Ding et al. "Improving Knowledge Graph Embedding Using Simple Constraints". In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*. Association for Computational Linguistics, 2018, pp. 110–121. DOI: 10.18653/v1/P18-1011.

[41] John Duchi, Elad Hazan, and Yoram Singer. "Adaptive Subgradient Methods for Online Learning and Stochastic Optimization". In: vol. 12. null. JMLR.org, July 2011, pp. 2121–2159.

[42] Takuma Ebisu and Ryutaro Ichise. "TorusE: Knowledge Graph Embedding on a Lie Group". In: *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*. AAAI'18/IAAI'18/EAAI'18. New Orleans, Louisiana, USA: AAAI Press, 2018. ISBN: 978-1-57735-800-8.

[43] Gavin Edwards et al. "Explainable Biomedical Recommendations via Reinforcement Learning Reasoning on Knowledge Graphs". In: 2021.

[44] Lisa Ehrlinger and Wolfram Wöß. "Towards a Definition of Knowledge Graphs". In: *Joint Proceedings of the Posters and Demos Track of the 12th International Conference on Semantic Systems - SEMANTiCS2016 and the 1st International Workshop on Semantic Change & Evolving Semantics (SuCCESS'16) co-located with the 12th International Conference on Semantic Systems (SEMANTiCS 2016), Leipzig, Germany, September 12-15, 2016*. Vol. 1695. CEUR Workshop Proceedings. CEUR-WS.org, 2016.

[45] Wolfgang Fahl et al. "Getting and hosting your own copy of Wikidata". In: *Proceedings of the 3rd Wikidata Workshop*. 2022.

[46] Matthew Fahrbach et al. "Faster Graph Embeddings via Coarsening". In: *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*. Vol. 119. Proceedings of Machine Learning Research. PMLR, 2020, pp. 2953–2963.

[47] Bahare Fatemi et al. “Knowledge Hypergraphs: Prediction Beyond Binary Relations”. In: *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*. Ed. by Christian Bessiere. 2020, pp. 2191–2197. DOI: 10.24963/IJCAI.2020/303.

[48] Luis Antonio Galárraga et al. “AMIE: association rule mining under incomplete evidence in ontological knowledge bases”. In: *22nd International World Wide Web Conference, WWW '13, Rio de Janeiro, Brazil, May 13-17, 2013*. International World Wide Web Conferences Steering Committee / ACM, 2013, pp. 413–422. DOI: 10.1145/2488388.2488425.

[49] Daniel Garijo and Mar'ia Poveda-Villal'on. “Best Practices for Implementing FAIR Vocabularies and Ontologies on the Web”. In: *Applications and Practices in Ontology Design, Extraction, and Reasoning*. 2020.

[50] Yuxia Geng et al. “Disentangled Ontology Embedding for Zero-shot Learning”. In: *KDD '22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 14 - 18, 2022*. ACM, 2022, pp. 443–453. DOI: 10.1145/3534678.3539453.

[51] Yuxia Geng et al. “OntoZSL: Ontology-enhanced Zero-shot Learning”. In: *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*. 2021, pp. 3325–3336. DOI: 10.1145/3442381.3450042.

[52] Yuxia Geng et al. “Relational Message Passing for Fully Inductive Knowledge Graph Completion”. In: *39th IEEE International Conference on Data Engineering, ICDE 2023, Anaheim, CA, USA, April 3-7, 2023*. IEEE, 2023, pp. 1221–1233. DOI: 10.1109/ICDE55515.2023.00098.

[53] Emma J. Gerritse, Faegheh Hasibi, and Arjen P. de Vries. “Graph-Embedding Empowered Entity Retrieval”. In: *Advances in Information Retrieval: 42nd European Conference on IR Research, ECIR 2020, Lisbon, Portugal, April 14–17, 2020, Proceedings, Part I*. Springer-Verlag, 2020, pp. 97–110. ISBN: 978-3-030-45438-8. DOI: 10.1007/978-3-030-45439-5_7.

[54] Birte Glimm et al. “HermiT: An OWL 2 Reasoner”. In: vol. 53. 3. 2014, pp. 245–269. DOI: 10.1007/s10817-014-9305-1.

[55] Xavier Glorot and Yoshua Bengio. “Understanding the difficulty of training deep feedforward neural networks”. In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. Vol. 9. Chia Laguna Resort, Sardinia, Italy: PMLR, 13–15 May 2010, pp. 249–256.

[56] Ian Goodfellow et al. “Generative Adversarial Networks”. In: vol. 3. June 2014. DOI: 10.1145/3422622.

[57] László Grad-Gyenge, Attila Kiss, and Peter Filzmoser. “Graph Embedding Based Recommendation Techniques on the Knowledge Graph”. In: *Proc. of the 25th Conf. on User Modeling, Adaptation and Personalization, UMAP*. 2017, pp. 354–359.

[58] Thomas R. Gruber. “Toward principles for the design of ontologies used for knowledge sharing?” In: vol. 43. 5-6. 1995, pp. 907–928. DOI: 10.1006/ijhc.1995.1081.

[59] Qingyu Guo et al. “A Survey on Knowledge Graph-Based Recommender Systems”. In: vol. 34. 8. 2022, pp. 3549–3568. DOI: 10.1109/TKDE.2020.3028705.

[60] Shu Guo et al. "Semantically Smooth Knowledge Graph Embedding". In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*. The Association for Computer Linguistics, 2015, pp. 84–94. DOI: 10.3115/v1/p15-1009.

[61] Ainaz Hajimoradlou and Mehran Kazemi. "Stay positive: Knowledge graph embedding without negative sampling". In: 2022.

[62] Armin Haller et al. "An Analysis of Links in Wikidata". In: *The Semantic Web: 19th International Conference, ESWC 2022, Hersonissos, Crete, Greece, May 29 – June 2, 2022, Proceedings*. Hersonissos, Greece: Springer-Verlag, 2022, pp. 21–38. ISBN: 978-3-031-06980-2. DOI: 10.1007/978-3-031-06981-9_2.

[63] Xu Han et al. "OpenKE: An Open Toolkit for Knowledge Embedding". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018: System Demonstrations, Brussels, Belgium, October 31 - November 4, 2018*. Association for Computational Linguistics, 2018, pp. 139–144. DOI: 10.18653/V1/D18-2024.

[64] Junheng Hao et al. "Universal Representation Learning of Knowledge Bases by Jointly Embedding Instances and Ontological Concepts". In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*. ACM, 2019, pp. 1709–1719. DOI: 10.1145/3292500.3330838.

[65] Steven Haussmann et al. "FoodKG: A Semantics-Driven Knowledge Graph for Food Recommendation". In: Oct. 2019, pp. 146–162. ISBN: 978-3-030-30795-0. DOI: 10.1007/978-3-030-30796-710.

[66] Remko Helms and Kees Buijsrogge. "Knowledge Network Analysis: A Technique to Analyze Knowledge Management Bottlenecks in Organizations". In: *16th International Workshop on Database and Expert Systems Applications (DEXA 2005), 22-26 August 2005, Copenhagen, Denmark*. IEEE Computer Society, 2005, pp. 410–414. DOI: 10.1109/DEXA.2005.127.

[67] Pascal Hitzler et al. "A Survey on Knowledge Graph Embeddings with Literals: Which Model Links Better Literal-Ly?" In: vol. 12. 4. NLD: IOS Press, Jan. 2021, pp. 617–647. DOI: 10.3233/SW-200404.

[68] Hlomani Hlomani and Deborah A. Stacey. "Approaches , methods , metrics , measures , and subjectivity in ontology evaluation : A survey". In: 2014.

[69] Aidan Hogan et al. "Knowledge Graphs". In: vol. 54. 4. Association for Computing Machinery (ACM), May 2022, pp. 1–37.

[70] Charles Tapley Hoyt et al. "A Unified Framework for Rank-based Evaluation Metrics for Link Prediction in Knowledge Graphs". In: 2022.

[71] Nicolas Hubert, Armelle Brun, and Davy Monticolo. "New Ontology and Knowledge Graph for University Curriculum Recommendation". In: *Proceedings of the ISWC 2022 Posters, Demos and Industry Tracks: From Novel Ideas to Industrial Practice co-located with 21st International Semantic Web Conference (ISWC 2022), Virtual Conference, Hangzhou, China, October 23-27, 2022*. Vol. 3254. CEUR Workshop Proceedings. 2022.

[72] Nicolas Hubert, Armelle Brun, and Davy Monticolo. "Vers un système de recommandation explicable pour l'orientation scolaire". In: *Workshop EXPLAIN'AI - EGC Blois 2022*. Blois, France, Jan. 2022. URL: https://hal.science/hal-03559471.

[73] Nicolas Hubert, Pierre Monnin, and Heiko Paulheim. "Beyond Transduction: A Survey on Inductive, Few Shot, and Zero Shot Link Prediction in Knowledge Graphs". In: 2023.

[74] Nicolas Hubert et al. *Do Similar Entities have Similar Embeddings?* 2024.

[75] Nicolas Hubert et al. "Knowledge Graph Embeddings for Link Prediction: Beware of Semantics!" In: *Proceedings of the Workshop on Deep Learning for Knowledge Graphs (DL4KG 2022) co-located with the 21th International Semantic Web Conference (ISWC 2022)*. Oct. 2022.

[76] Nicolas Hubert et al. "New Strategies for Learning Knowledge Graph Embeddings: The Recommendation Case". In: *Knowledge Engineering and Knowledge Management - 23rd International Conference, EKAW 2022, Bolzano, Italy, September 26-29, 2022, Proceedings*. Vol. 13514. Lecture Notes in Computer Science. Springer, 2022, pp. 66–80. DOI: 10.1007/978-3-031-17105-5_5.

[77] Nicolas Hubert et al. "PyGraft: Configurable Generation of Synthetic Schemas and Knowledge Graphs at Your Fingertips". In: *The Semantic Web - 21st International Conference, ESWC 2024, Hersonissos, Crete, Greece, May 26 - May 30, 2024, Proceedings*. Lecture Notes in Computer Science. Springer, 2024.

[78] Nicolas Hubert et al. "Schema First! Learn Versatile Knowledge Graph Embeddings by Capturing Semantics with MASCHInE". In: *K-CAP '23: Knowledge Capture Conference, Pensacola, Florida, USA, December 5-7, 2023*. ACM, 2023. DOI: 10.48550/ARXIV.2306.03659.

[79] Nicolas Hubert et al. "Sem@K: Is my knowledge graph embedding model semantic-aware?" In: vol. 14. Dec. 2023, pp. 1–37. DOI: 10.3233/SW-233508.

[80] Nicolas Hubert et al. "Treat Different Negatives Differently: Enriching Loss Functions with Domain and Range Constraints for Link Prediction". In: *The Semantic Web - 21st International Conference, ESWC 2024, Hersonissos, Crete, Greece, May 26 - May 30, 2024, Proceedings*. Lecture Notes in Computer Science. Springer, 2024.

[81] Mohammed Essmat Ibrahim, Yanyan Yang, and David Ndzi. "Using Ontology for Personalised Course Recommendation Applications". In: *Computational Science and Its Applications – ICCSA 2017*. Springer International Publishing, 2017, pp. 426–438. DOI: 10.1007/978-3-319-62392-4_31.

[82] Mohammed Essmat Ibrahim et al. "Ontology-Based Personalized Course Recommendation Framework". In: vol. 7. 2019, pp. 5180–5199.

[83] Eleni Ilkou et al. "EduCOR: An Educational and Career-Oriented Recommendation Ontology". In: Springer International Publishing, 2021, pp. 546–562. ISBN: 9783030883614. DOI: 10.1007/978-3-030-88361-4_32.

[84] Md Kamrul Islam, Sabeur Aridhi, and Malika Smail-Tabbone. "Negative sampling and rule mining for explainable link prediction in knowledge graphs". In: vol. 250. 2022.

[85] Tatyana Ivanova and Miroslav Popov. "Ontology Evaluation and Multilingualism". In: *Proceedings of the 21st International Conference on Computer Systems and Technologies '20*. Association for Computing Machinery, 2020, pp. 215–222. ISBN: 9781450377683.

[86] Nitisha Jain et al. “Do Embeddings Actually Capture Knowledge Graph Semantics?” In: *The Semantic Web - 18th International Conference, ESWC*. Vol. 12731. LNCS. Springer, 2021, pp. 143–159.

[87] Nitisha Jain et al. “Improving Knowledge Graph Embeddings with Ontological Reasoning”. In: *The Semantic Web - ISWC 2021 - 20th International Semantic Web Conference, ISWC 2021, Virtual Event, October 24-28, 2021, Proceedings*. Vol. 12922. Lecture Notes in Computer Science. Springer, 2021, pp. 410–426. DOI: 10.1007/978-3-030-88361-4_24.

[88] Guoliang Ji et al. “Knowledge Graph Embedding via Dynamic Mapping Matrix”. In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*. The Association for Computer Linguistics, 2015, pp. 687–696. DOI: 10.3115/v1/p15-1067.

[89] Shaoxiong Ji et al. “A Survey on Knowledge Graphs: Representation, Acquisition, and Applications”. In: vol. 33. 2. 2022, pp. 494–514. DOI: 10.1109/TNNLS.2021.3070843.

[90] Shaoxiong Ji et al. “A Survey on Knowledge Graphs: Representation, Acquisition, and Applications”. In: vol. 33. 2. 2022, pp. 494–514. DOI: 10.1109/TNNLS.2021.3070843.

[91] Long Jin et al. “A Comprehensive Study on Knowledge Graph Embedding over Relational Patterns Based on Rule Learning”. In: *The Semantic Web - ISWC 2023 - 22nd International Semantic Web Conference, Athens, Greece, November 6-10, 2023, Proceedings, Part I*. Vol. 14265. Lecture Notes in Computer Science. Springer, 2023, pp. 290–308. DOI: 10.1007/978-3-031-47240-4_16.

[92] Evangelos Katis et al. “Developing an Ontology for Curriculum and Syllabus: ESWC 2018 Satellite Events, Heraklion, Crete, Greece, June 3-7, 2018, Revised Selected Papers”. In: Aug. 2018, pp. 55–59. ISBN: 978-3-319-98191-8. DOI: 10.1007/978-3-319-98192-5_11.

[93] Seyed Mehran Kazemi and David Poole. “SimplE Embedding for Link Prediction in Knowledge Graphs”. In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*. 2018, pp. 4289–4300.

[94] Diederik P. Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization”. In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. 2015.

[95] Yasamin Klingler et al. “Evaluation of algorithms for interaction-sparse recommendations : neural networks don’t always win”. en. In: OpenProceedings, 2022. DOI: 10.21256/ZHAW-24616.

[96] Bhushan Kotnis and Vivi Nastase. “Analysis of the impact of negative sampling on link prediction in knowledge graphs”. In: 2017.

[97] Denis Krompaß, Stephan Baier, and Volker Tresp. “Type-Constrained Representation Learning in Knowledge Graphs”. In: *The Semantic Web - 14th International Semantic Web Conference (ISWC)*. Vol. 9366. Springer, 2015, pp. 640–655.

[98] Peter Kümmel. "An Algorithm of Limited Syntax Based on Language Universals". In: *Computational And Mathematical Linguistics: Proceedings of the 5th International Conference on Computational Linguistics, COLING 1973, Pisa, Italy, August 27 - September 1, 1973*. Ed. by Antonio Zampolli and Nicoletta Calzolari. ACL, 1973, pp. 225–248. URL: https://aclanthology.org/C73-2020/.

[99] Jonathan Lajus, Luis Galárraga, and Fabian M. Suchanek. "Fast and Exact Rule Mining with AMIE 3". In: *The Semantic Web - 17th International Conference, ESWC 2020, Heraklion, Crete, Greece, May 31-June 4, 2020, Proceedings*. Vol. 12123. Lecture Notes in Computer Science. Springer, 2020, pp. 36–52. DOI: 10.1007/978-3-030-49461-2_3.

[100] Claudia Leacock and Martin Chodorow. "Combining Local Context and WordNet Similarity for Word Sense Identification". In: vol. 49. Jan. 1998, pp. 265–.

[101] Patrick S. H. Lewis et al. "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks". In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. 2020.

[102] Yuhua Li, Zuhair Bandar, and David McLean. "An Approach for Measuring Semantic Similarity between Words Using Multiple Information Sources". In: vol. 15. 4. 2003, pp. 871–882. DOI: 10.1109/TKDE.2003.1209005.

[103] Zelong Li et al. "Efficient Non-Sampling Knowledge Graph Embedding". In: *Proceedings of the Web Conference 2021*. Ljubljana, Slovenia: Association for Computing Machinery, 2021, pp. 1727–1736. ISBN: 9781450383127. DOI: 10.1145/3442381.3449859.

[104] Wenqing Lin et al. "Initialization for Network Embedding: A Graph Partition Approach". In: *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining, Houston, TX, USA, February 3-7, 2020*. ACM, 2020, pp. 367–374. DOI: 10.1145/3336191.3371781.

[105] Yankai Lin et al. "Learning Entity and Relation Embeddings for Knowledge Graph Completion". In: *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*. AAAI Press, 2015, pp. 2181–2187.

[106] Zhenzhou Lin et al. "Hierarchical Type Enhanced Negative Sampling for Knowledge Graph Embedding". In: *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*. ACM, 2023, pp. 2047–2051. DOI: 10.1145/3539618.3591996.

[107] Xin Lv et al. "Differentiating Concepts and Instances for Knowledge Graph Embedding". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*. Association for Computational Linguistics, 2018, pp. 1971–1979. DOI: 10.18653/v1/d18-1222.

[108] Farzaneh Mahdisoltani, Joanna Biega, and Fabian M. Suchanek. "YAGO3: A Knowledge Base from Multilingual Wikipedias". In: *Seventh Biennial Conference on Innovative Data Systems Research, CIDR 2015, Asilomar, CA, USA, January 4-7, 2015, Online Proceedings*. 2015.

[109] Tomás Mikolov et al. "Efficient Estimation of Word Representations in Vector Space". In: *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*. 2013.

[110] George A. Miller. "WordNet: A Lexical Database for English". In: vol. 38. 11. 1995, pp. 39–41.

[111] Pasquale Minervini et al. "Regularizing Knowledge Graph Embeddings via Equivalence and Inversion Axioms". In: *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2017, Skopje, Macedonia, September 18-22, 2017, Proceedings, Part I*. Vol. 10534. Lecture Notes in Computer Science. Springer, 2017, pp. 668–683. DOI: 10.1007/978-3-319-71249-9_40.

[112] T. Mitchell et al. "Never-Ending Learning". In: vol. 61. 5. New York, NY, USA: Association for Computing Machinery, Apr. 2018, pp. 103–115. DOI: 10.1145/3191513.

[113] Aisha Mohamed et al. "Popularity Agnostic Evaluation of Knowledge Graph Embeddings". In: *Proceedings of the Thirty-Sixth Conference on Uncertainty in Artificial Intelligence, UAI 2020, virtual online, August 3-6, 2020*. Vol. 124. Proceedings of Machine Learning Research. AUAI Press, 2020, pp. 1059–1068.

[114] Sameh K. Mohamed, Emir Muñoz, and Vit Novacek. "On Training Knowledge Graph Embedding Models". In: vol. 12. 4. 2021. DOI: 10.3390/info12040147.

[115] Sameh K. Mohamed et al. "Loss Functions in Knowledge Graph Embedding Models". In: *Proceedings of the Workshop on Deep Learning for Knowledge Graphs (DL4KG2019) Co-located with the 16th Extended Semantic Web Conference 2019 (ESWC 2019), Portoroz, Slovenia, June 2, 2019*. Vol. 2377. CEUR Workshop Proceedings. CEUR-WS.org, 2019, pp. 1–10.

[116] Changsung Moon, Paul Jones, and Nagiza F. Samatova. "Learning Entity Type Embeddings for Knowledge Graph Completion". In: *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. CIKM '17. Singapore, Singapore: Association for Computing Machinery, 2017, pp. 2215–2218. ISBN: 9781450349185. DOI: 10.1145/3132847.3133095.

[117] Shaimaa M Nafea, François Siewe, and Ying He. "On recommendation of learning objects using felder-silverman learning style model". In: *IEEE Access* 7 (2019), pp. 163034–163048.

[118] Deepak Nathani et al. "Learning Attention-based Embeddings for Relation Prediction in Knowledge Graphs". In: *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*. Association for Computational Linguistics, 2019, pp. 4710–4723. DOI: 10.18653/v1/p19-1466.

[119] Roberto Navigli and Simone Paolo Ponzetto. "BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network". In: vol. 193. 2012, pp. 217–250. DOI: 10.1016/J.ARTINT.2012.07.001. URL: https://doi.org/10.1016/j.artint.2012.07.001.

[120] Dai Quoc Nguyen et al. "A Novel Embedding Model for Knowledge Base Completion Based on Convolutional Neural Network". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers)*. Association for Computational Linguistics, 2018, pp. 327–333. DOI: 10.18653/v1/n18-2053.

[121] Maximilian Nickel, Lorenzo Rosasco, and Tomaso A. Poggio. "Holographic Embeddings of Knowledge Graphs". In: *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*. AAAI Press, 2016, pp. 1955–1961.

[122] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. "A Three-Way Model for Collective Learning on Multi-Relational Data". In: *Proc. of the 28th International Conference on Machine Learning, ICML*. 2011, pp. 809–816.

[123] Maximilian Nickel et al. "A Review of Relational Machine Learning for Knowledge Graphs". In: vol. 104. 1. 2016, pp. 11–33. DOI: 10.1109/JPROC.2015.2483592.

[124] Natalya F Noy and Deborah L McGuinness. "Ontology development 101: A guide to creating your first ontology". In: Stanford Knowledge Systems Laboratory and Stanford Medical Informatics, 2001.

[125] Charbel Obeid et al. "Ontology-Based Recommender System in Higher Education". In: *Companion Proceedings of the The Web Conference 2018*. WWW '18. Lyon, France: International World Wide Web Conferences Steering Committee, 2018, pp. 1031–1034. ISBN: 9781450356404. DOI: 10.1145/3184558.3191533.

[126] Enrico Palumbo et al. "Translational Models for Item Recommendation". In: *The Semantic Web: European Semantic Web Conf. ESWC*. Vol. 11155. 2018, pp. 478–490.

[127] Zachary A. Pardos and Weijie Jiang. "Designing for Serendipity in a University Course Recommendation System". In: *Proceedings of the Tenth International Conference on Learning Analytics amp; Knowledge*. LAK '20. Frankfurt, Germany: Association for Computing Machinery, 2020, pp. 350–359. ISBN: 9781450377126. DOI: 10.1145/3375462.3375524.

[128] Archit Parnami et al. "Transformation of Node to Knowledge Graph Embeddings for Faster Link Prediction in Social Networks". In: *Proceedings of the 3rd International Workshop on Knowledge Graph Construction (KGCW 2022) co-located with 19th Extended Semantic Web Conference (ESWC 2022), Hersonissos, Greek, May 30, 2022*. Vol. 3141. CEUR Workshop Proceedings. 2022.

[129] Heiko Paulheim. "Knowledge graph refinement: A survey of approaches and evaluation methods". In: vol. 8. 2017, pp. 489–508.

[130] Maria Angela Pellegrino et al. "GEval: A Modular and Extensible Evaluation Framework for Graph Embedding Techniques". In: *The Semantic Web - 17th International Conference, ESWC 2020, Heraklion, Crete, Greece, May 31-June 4, 2020, Proceedings*. Vol. 12123. Lecture Notes in Computer Science. Springer, 2020, pp. 565–582.

[131] Thomas Pellissier Tanon, Gerhard Weikum, and Fabian Suchanek. "YAGO 4: A Reason-Able Knowledge Base". In: *The Semantic Web: 17th International Conference, ESWC 2020, Heraklion, Crete, Greece, May 31–June 4, 2020, Proceedings*. Heraklion, Crete, Greece: Springer-Verlag, 2020, pp. 583–596. ISBN: 978-3-030-49460-5. DOI: 10.1007/978-3-030-49461-2_34.

[132] Xutan Peng et al. "Highly Efficient Knowledge Graph Embedding Learning with Orthogonal Procrustes Analysis". In: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Online: Association for Computational Linguistics, June 2021, pp. 2364–2375. DOI: 10.18653/v1/2021.naacl-main.187.

[133] Jeffrey Pennington, Richard Socher, and Christopher Manning. "GloVe: Global Vectors for Word Representation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1532–1543. DOI: 10.3115/v1/D14-1162.

[134] Marcin Pietrasik and Marek Z. Reformat. "Probabilistic Coarsening for Knowledge Graph Embeddings". In: vol. 12. 3. 2023, p. 275.

[135] Alessandro Piscopo and Elena Simperl. "Who Models the World? Collaborative Ontology Creation and User Roles in Wikidata". In: vol. 2. CSCW. Association for Computing Machinery, Nov. 2018. DOI: 10.1145/3274410.

[136] Alessandro Piscopo et al. "Provenance Information in a Collaborative Knowledge Graph: An Evaluation of Wikidata External References". In: *The Semantic Web – ISWC 2017: 16th International Semantic Web Conference, Vienna, Austria, October 21–25, 2017, Proceedings, Part I*. Vienna, Austria: Springer-Verlag, 2017, pp. 542–558. ISBN: 978-3-319-68287-7. DOI: 10.1007/978-3-319-68288-4_32.

[137] Jan Portisch, Michael Hladik, and Heiko Paulheim. "RDF2Vec Light - A Lightweight Approachfor Knowledge Graph Embeddings". In: *Proceedings of the ISWC 2020 Demos and Industry Tracks: From Novel Ideas to Industrial Practice co-located with 19th International Semantic Web Conference (ISWC 2020), Globally online, November 1-6, 2020 (UTC)*. Vol. 2721. CEUR Workshop Proceedings. CEUR-WS.org, 2020, pp. 79–84.

[138] Jan Portisch and Heiko Paulheim. "The DLCC Node Classification Benchmark for Analyzing Knowledge Graph Embeddings". In: *The Semantic Web - ISWC 2022 - 21st International Semantic Web Conference, Virtual Event, October 23-27, 2022, Proceedings*. Vol. 13489. Lecture Notes in Computer Science. Springer, 2022, pp. 592–609. DOI: 10.1007/978-3-031-19433-7_34.

[139] Joe Raad and Christophe Cruz. "A Survey on Ontology Evaluation Methods". In: *Proceedings of the International Conference on Knowledge Engineering and Ontology Development, part of the 7th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management*. Lisbonne, Portugal, Nov. 2015. DOI: 10.5220/0005591001790186.

[140] Roy Rada et al. "Development and application of a metric on semantic nets". In: vol. 19. 1. 1989, pp. 17–30. DOI: 10.1109/21.24528.

[141] Thomas Rebele et al. "YAGO: A Multilingual Knowledge Base from Wikipedia, Wordnet, and Geonames". In: *International Workshop on the Semantic Web*. 2016.

[142] Philip Resnik. "Semantic Similarity in a Taxonomy: An Information-Based Measure and its Application to Problems of Ambiguity in Natural Language". In: vol. 11. 1999, pp. 95–130. DOI: 10.1613/jair.514.

[143] Andrea Rossi et al. "Explaining Link Prediction Systems Based on Knowledge Graph Embeddings". In: *Proceedings of the 2022 International Conference on Management of Data*. SIGMOD '22. Philadelphia, PA, USA: Association for Computing Machinery, 2022, pp. 2062–2075. ISBN: 9781450392495. DOI: 10.1145/3514221.3517887.

[144] Andrea Rossi et al. "Knowledge Graph Embedding for Link Prediction: A Comparative Analysis". In: vol. 15. 2. 2021, 14:1–14:49.

[145] Andrea Rossi et al. "Knowledge Graph Embedding for Link Prediction: A Comparative Analysis". In: vol. 15. 2. 2021, 14:1–14:49.

[146] Tara Safavi and Danai Koutra. "CoDEx: A Comprehensive Knowledge Graph Completion Benchmark". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*. Association for Computational Linguistics, 2020, pp. 8328–8350. DOI: 10.18653/v1/2020.emnlp-main.669.

[147] Michael Sejr Schlichtkrull et al. "Modeling Relational Data with Graph Convolutional Networks". In: *The Semantic Web - 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings*. Vol. 10843. Lecture Notes in Computer Science. Springer, 2018, pp. 593–607. DOI: 10.1007/978-3-319-93417-4_38.

[148] Edward W Schneider. "Course Modularization Applied: The Interface System and Its Implications For Sequence Control and Data Analysis." In: ERIC, 1973.

[149] Chao Shang et al. "End-to-End Structure-Aware Convolutional Networks for Knowledge Base Completion". In: *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*. AAAI Press, 2019, pp. 3060–3067. DOI: 10.1609/aaai.v33i01.33013060.

[150] Baoxu Shi and Tim Weninger. "Open-World Knowledge Graph Completion". In: *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*. AAAI Press, 2018, pp. 1957–1964. DOI: 10.1609/AAAI.V32I1.11535.

[151] Amit Singhal. *Introducing the Knowledge Graph: things, not strings*. 2020-11-13. 2012. URL: https://www.blog.google/products/search/introducing-knowledge-graph-things-not/.

[152] Shamane Siriwardhana et al. "Improving the Domain Adaptation of Retrieval Augmented Generation (RAG) Models for Open Domain Question Answering". In: vol. 11. 2023, pp. 1–17. DOI: 10.1162/tacl_a_00530.

[153] John F. Sowa. "Semantics of Conceptual Graphs". In: *17th Annual Meeting of the Association for Computational Linguistics, 29 June - 1 July 1979, University of California at San Diego, La Jolla, CA, USA*. Ed. by Norman K. Sondheimer. ACL, 1979. DOI: 10.3115/982163.982175.

[154] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. "Yago: A Core of Semantic Knowledge". In: *Proceedings of the 16th International Conference on World Wide Web*. WWW '07. Banff, Alberta, Canada: Association for Computing Machinery, 2007, pp. 697–706. ISBN: 9781595936547. DOI: 10.1145/1242572.1242667.

[155] Yizhou Sun et al. "PathSim: Meta Path-Based Top-K Similarity Search in Heterogeneous Information Networks". In: vol. 4. 11. 2011, pp. 992–1003. URL: http://www.vldb.org/pvldb/vol4/p992-sun.pdf.

[156] Zhiqing Sun et al. "A Re-evaluation of Knowledge Graph Completion Methods". In: *Proc. of the 58th Annual Meeting of the Association for Computational Linguistics ACL*. Association for Computational Linguistics, 2020, pp. 5516–5522.

[157] Zhiqing Sun et al. "RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space". In: *7th International Conference on Learning Representations, ICLR*. 2019.

[158] Thomas Pellissier Tanon, Gerhard Weikum, and Fabian M. Suchanek. "YAGO 4: A Reason-able Knowledge Base". In: *The Semantic Web - 17th International Conference, ESWC 2020, Heraklion, Crete, Greece, May 31-June 4, 2020, Proceedings*. Vol. 12123. Lecture Notes in Computer Science. Springer, 2020, pp. 583–596. DOI: 10.1007/978-3-030-49461-2_34.

[159] Thomas Pellissier Tanon et al. "From Freebase to Wikidata: The Great Migration". In: *Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada, April 11 - 15, 2016*. ACM, 2016, pp. 1419–1428. DOI: 10.1145/2872427.2874809.

[160] Andon Tchechmedjiev et al. "ClaimsKG: A Knowledge Graph of Fact-Checked Claims". In: *The Semantic Web - ISWC 2019 - 18th International Semantic Web Conference, Auckland, New Zealand, October 26-30, 2019, Proceedings, Part II*. Vol. 11779. Lecture Notes in Computer Science. Springer, 2019, pp. 309–324. DOI: 10.1007/978-3-030-30796-7_20.

[161] Tijmen Tieleman and Geoffrey Hinton. "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude". In: vol. 4. 2. 2012, pp. 26–31.

[162] Sudhanshu Tiwari, Iti Bansal, and Carlos R. Rivero. "Revisiting the Evaluation Protocol of Knowledge Graph Completion Methods for Link Prediction". In: *WWW '21: The Web Conference*. ACM / IW3C2, 2021, pp. 809–820.

[163] Kristina Toutanova and Danqi Chen. "Observed versus latent features for knowledge base and text inference". In: *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality, CVSC 2015, Beijing, China, July 26-31, 2015*. Association for Computational Linguistics, 2015, pp. 57–66. DOI: 10.18653/v1/W15-4007.

[164] Théo Trouillon et al. "Complex Embeddings for Simple Link Prediction". In: *Proceedings of the 33rd International Conference on Machine Learning, ICML*. Vol. 48. 2016, pp. 2071–2080.

[165] Théo Trouillon et al. "Complex Embeddings for Simple Link Prediction". In: *Proc. of the 33rd International Conf. on Machine Learning, ICML*. Vol. 48. 2016, pp. 2071–2080.

[166] Shikhar Vashishth et al. "Composition-based Multi-Relational Graph Convolutional Networks". In: *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[167] Denny Vrandecic and Markus Krötzsch. "Wikidata: a free collaborative knowledgebase". In: vol. 57. 10. 2014, pp. 78–85.

[168] Denny Vrandečić, Lydia Pintscher, and Markus Krötzsch. "Wikidata: The Making Of". In: *Companion Proceedings of the ACM Web Conference 2023*. WWW '23 Companion. Austin, TX, USA: Association for Computing Machinery, 2023, pp. 615–624. ISBN: 9781450394192. DOI: 10.1145/3543873.3585579.

[169] Haoyu Wang et al. "A Lightweight Knowledge Graph Embedding Framework for Efficient Inference and Storage". In: *Proceedings of the 30th ACM International Conference on Information and Knowledge Management*. CIKM '21. Virtual Event, Queensland, Australia: Association for Computing Machinery, 2021, pp. 1909–1918. ISBN: 9781450384469. DOI: 10.1145/3459637.3482224.

[170] Meihong Wang, Linling Qiu, and Xiaoli Wang. "A Survey on Knowledge Graph Embeddings for Link Prediction". In: vol. 13. 3. 2021, p. 485.

[171] Quan Wang et al. “Knowledge Graph Embedding: A Survey of Approaches and Applications”. In: vol. 29. 12. 2017, pp. 2724–2743.

[172] Xiangmeng Wang et al. *Causal Disentanglement for Semantic-Aware Intent Learning in Recommendation*. 2023. DOI: 10.1109/TKDE.2022.3159802.

[173] Yanjie Wang et al. “On Evaluating Embedding Models for Knowledge Base Completion”. In: *Proc. of the 4th Workshop on Representation Learning for NLP, RepL4NLP@ACL*. 2019, pp. 104–112.

[174] Zhen Wang et al. “Knowledge Graph Embedding by Translating on Hyperplanes”. In: *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*. 2014, pp. 1112–1119.

[175] Michael Weyns et al. “Conditional Constraints for Knowledge Graph Embeddings”. In: *Proc. of the Workshop on Deep Learning for Knowledge Graphs (DL4KG@ISWC)*. Vol. 2635. 2020.

[176] David S. Wishart et al. “DrugBank: a knowledgebase for drugs, drug actions and drug targets”. In: vol. 36. Database-Issue. 2008, pp. 901–906. DOI: 10.1093/NAR/GKM958.

[177] Zhibiao Wu and Martha Palmer. “Verbs Semantics and Lexical Selection”. In: *Proceedings of the 32nd Annual Meeting on Association for Computational Linguistics*. ACL ’94. Las Cruces, New Mexico: Association for Computational Linguistics, 1994, pp. 133–138. DOI: 10.3115/981732.981751.

[178] Han Xiao, Minlie Huang, and Xiaoyan Zhu. “TransG : A Generative Model for Knowledge Graph Embedding”. In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics, 2016. DOI: 10.18653/v1/p16-1219.

[179] Ruobing Xie, Zhiyuan Liu, and Maosong Sun. “Representation Learning of Knowledge Graphs with Hierarchical Types”. In: *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*. IJCAI/AAAI Press, 2016, pp. 2965–2971.

[180] Gongwen Xu et al. “Personalized Course Recommendation System Fusing with Knowledge Graph and Collaborative Filtering”. In: vol. 2021. 2021, p. 8.

[181] Bishan Yang et al. “Embedding Entities and Relations for Learning and Inference in Knowledge Bases”. In: *3rd International Conference on Learning Representations, ICLR*. 2015.

[182] Shih-Yuan Yu et al. “Pykg2vec: A Python Library for Knowledge Graph Embedding”. In: vol. 22. 2021, 16:1–16:6.

[183] Shichang Zhang et al. “PaGE-Link: Path-based Graph Neural Network Explanation for Heterogeneous Link Prediction”. In: 2023.

[184] Wen Zhang et al. “NeuralKG: An Open Source Library for Diverse Representation Learning of Knowledge Graphs”. In: *SIGIR ’22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*. ACM, 2022, pp. 3323–3328. DOI: 10.1145/3477495.3531669.

[185] Wen Zhang et al. “XTransE: Explainable Knowledge Graph Embedding for Link Prediction with Lifestyles in e-Commerce”. In: Feb. 2020, pp. 78–87. ISBN: 978-981-15-3411-9. DOI: 10.1007/978-981-15-3412-6_8.

[186] Yongqi Zhang et al. "NSCaching: Simple and Efficient Negative Sampling for Knowledge Graph Embedding". In: *35th IEEE International Conference on Data Engineering, ICDE 2019, Macao, China, April 8-11, 2019*. IEEE, 2019, pp. 614–625. DOI: 10.1109/ICDE.2019.00061.

[187] Yuan Zhang et al. *Distilling Structured Knowledge into Embeddings for Explainable and Accurate Recommendation*. Houston, TX, USA, 2020. DOI: 10.1145/3336191.3371790.

[188] Zhanqiu Zhang et al. "Learning Hierarchy-Aware Knowledge Graph Embeddings for Link Prediction". In: *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*. AAAI Press, 2020, pp. 3065–3072.

[189] Da Zheng et al. "DGL-KE: Training Knowledge Graph Embeddings at Scale". In: *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '20. New York, NY, USA: Association for Computing Machinery, 2020, pp. 739–748.

[190] Zhaocheng Zhu et al. "Neural bellman-ford networks: A general graph neural network framework for link prediction". In: *NeurIPS*. 2021.

[191] Amal Zouaq and Félix Martel. "What is the schema of your knowledge graph?: leveraging knowledge graph embeddings and clustering for expressive taxonomy learning". In: *Proceedings of The International Workshop on Semantic Big Data, SBD@SIGMOD 2020, Portland, Oregon, USA, June 19, 2020*. ACM, 2020, 6:1–6:6.

Résumé

Les modèles d'embeddings à base de graphes de connaissances ont considérablement gagné en popularité ces dernières années. Ces modèles apprennent une représentation vectorielle des entités et des relations des graphes de connaissances (GCs). Cette thèse explore spécifiquement le progrès de tels modèles pour la tâche de prédiction de lien (PL), qui est d'une importance capitale car elle se retrouve dans plusieurs applications telles que les systèmes de recommandation. Dans cette thèse, divers défis liés à l'utilisation des modèles d'embeddings de GCs pour la PL sont identifiés : la rareté des ressources sémantiquement riches, la nature unidimensionnelle des cadres d'évaluation, et le manque de considérations sémantiques dans les approches d'apprentissage automatique. Cette thèse propose des solutions novatrices à ces défis. Premièrement, elle contribue au développement de ressources sémantiquement riches : les jeux de données principaux pour la prédiction de lien sont enrichis en utilisant des informations basées sur des schémas, EducOnto et EduKG sont proposés pour surmonter la pénurie de ressources dans le domaine éducatif, et PyGraft est introduit comme un outil innovant pour générer des ontologies synthétiques et des graphes de connaissances. Deuxièmement, la thèse propose une nouvelle métrique d'évaluation orientée sémantique, Sem@K, offrant une perspective multidimensionnelle sur la performance des modèles. Il est important de souligner que les modèles populaires sont réévalués en utilisant Sem@K, ce qui révèle des aspects essentiels et jusqu'alors inexplorés de leurs capacités respectives et souligne le besoin de cadres d'évaluation multidimensionnels. Troisièmement, la thèse se penche sur le développement d'approches neuro-symboliques, transcendant les paradigmes traditionnels de l'apprentissage automatique. Ces approches ne démontrent pas seulement une meilleure capacité sémantique dans leurs prédictions, mais étendent également leur utilité à diverses applications telles que les systèmes de recommandation. En résumé, le présent travail ne redéfinit pas seulement l'évaluation et la fonctionnalité des modèles d'embeddings de GCs, mais prépare également le terrain pour des systèmes d'intelligence artificielle plus polyvalents et interprétables, soutenant les explorations futures à l'intersection de l'apprentissage automatique et du raisonnement symbolique.

Mots-clés: Graphe de connaissance, Plongement de graphe, IA neuro-symbolique, Prédiction de lien

Abstract

Knowledge graph embedding models (KGEMs) have gained considerable traction in recent years. These models learn a vector representation of knowledge graph entities and relations, a.k.a. knowledge graph embeddings (KGEs). This thesis specifically explores the advancement of KGEMs for the link prediction (LP) task, which is of utmost importance as it underpins several downstream applications such as recommender systems. In this thesis, various challenges around the use of KGEMs for LP are identified: the scarcity of semantically rich resources, the unidimensional nature of evaluation frameworks, and the lack of semantic considerations in prevailing machine learning-based approaches. Central to this thesis is the proposition of novel solutions to these challenges. Firstly, the thesis contributes to the development of semantically rich resources: mainstream datasets for link prediction are enriched using schema-based information, EducOnto and EduKG are proposed to overcome the paucity of resources in the educational

domain, and PyGraft is introduced as an innovative open-source tool for generating synthetic ontologies and knowledge graphs. Secondly, the thesis proposes a new semantic-oriented evaluation metric, Sem@K, offering a multi-dimensional perspective on model performance. Importantly, popular models are reassessed using Sem@K, which reveals essential insights into their respective capabilities and highlights the need for multi-faceted evaluation frameworks. Thirdly, the thesis delves into the development of neuro-symbolic approaches, transcending traditional machine learning paradigms. These approaches do not only demonstrate improved semantic awareness but also extend their utility to diverse applications such as recommender systems. In summary, the present work not only redefines the evaluation and functionality of knowledge graph embedding models but also sets the stage for more versatile, interpretable AI systems, underpinning future explorations at the intersection of machine learning and symbolic reasoning.

Keywords: Knowledge graph, Graph embedding, Neuro-symbolic AI, Link prediction