Multi-GAT semi-supervisé pour l'extraction d'informations et son adaptation au chiffrement homomorphe

Djedjiga Belhadj

UNIVERSITÉ DE LORRAINE

École doctorale IAEM Lorraine

Multi-GAT semi-supervisé pour l'extraction d'informations et son adaptation au chiffrement homomorphe

THÈSE

présentée et soutenue publiquement le 28 mars 2024

pour l'obtention du

Doctorat de l'Université de Lorraine

(mention informatique)

par

Djedjiga Belhadj

Composition du jury

Président :	Claire Gardent	CNRS - LORIA
Rapporteurs :	Rolf Ingold	Université de Fribourg
	Jean-Yves Ramel	LIFAT - Université de Tours
Examinateurs :	Claire Gardent	CNRS - LORIA
	Véronique Eglin	LIRIS - INSA - CNRS
Invité :	Sonia Belaïd	CryptoExperts
Directeur de thèse :	Abdel Belaïd	Université de Lorraine - LORIA

Laboratoire Lorrain de Recherche en Informatique et ses Applications — UMR 7503

Mis en page avec la classe thesul.

Remerciements

Tout d'abord, je tiens à exprimer ma sincère gratitude à mon directeur de thèse, Monsieur Abdel Belaïd, pour son encadrement, ses précieux conseils, son aide, sa gentillesse et sa constante disponibilité. Ses conseils et ses encouragements ont été pour moi une grande source de motivation tout au long de l'élaboration de cette thèse. Ce fut un plaisir de travailler avec lui. Je lui suis très reconnaissante pour son soutien tout au long de ces années de thèse.

Je souhaite remercier chaleureusement Mme Yolande Belaïd pour l'aide précieuse qu'elle m'a apportée tout au long de ma thèse. Ses conseils, relectures, corrections et commentaires ont toujours été très pertinents et efficaces.

Je tiens à remercier sincèrement Monsieur Jean-Yves Ramel, Professeur à l'Université de Tours, et Monsieur Rolf Ingold, Professeur à l'Université de Fribourg, pour l'honneur qu'ils m'ont fait en acceptant d'être rapporteurs de ma thèse.

Je remercie également Madame Claire Gardant, directrice de recherche au CNRS, et Madame Véronique Eglin, professeur à l'INSA de Lyon, d'avoir accepté de participer à mon jury de thèse en tant qu'examinatrices et d'avoir pris le temps de lire mon rapport de thèse.

Je voudrais aussi remercier Sonia Belaïd, ingénieure chez CryptoExperts, qui a été invitée à participer au jury. Ce fut un réel plaisir de travailler avec elle durant les différentes étapes de ma thèse, en particulier durant la dernière partie. Grâce à elle et à toute l'équipe de CryptoExperts, j'ai pu explorer dans ma thèse un nouveau domaine lié à la sécurité et au chiffrement.

Je remercie Fair&Smart, l'entreprise porteuse du projet BPI DeepTech dans lequel s'inscrit ma thèse. Grâce à Fair&Smart, j'ai pu travailler sur cette thèse et y apporter plusieurs contributions.

Enfin, je tiens à dire un grand merci à ma famille et à mes amis qui m'ont soutenu pendant mes années de thèse, ainsi qu'à ceux qui étaient présents lors de ma soutenance.

Je dédie cette thèse
à l'âme de ma chère maman
« Fatima »,
qui nous a toujours inculqué
l'importance des études et de la science.

Résumé

Cette thèse est réalisée dans le cadre du projet BPI DeepTech, en collaboration avec la société Fair&Smart, veillant principalement à la protection des données personnelles conformément au Règlement Général sur la Protection des Données (RGPD). Dans ce contexte, nous avons proposé un modèle neuronal profond pour l'extraction d'informations dans les documents administratifs semi-structurés (DSSs). En raison du manque de données d'entraînement publiques, nous avons proposé un générateur artificiel de DSSs qui peut générer plusieurs classes de documents avec une large variation de contenu et de mise en page. Les documents sont générés à l'aide de variables aléatoires permettant de gérer le contenu et la mise en page en respectant des contraintes visant à garantir leur proximité avec des documents réels. Des métriques ont été introduites pour évaluer la diversité des DSSs générés en termes de contenu et de mise en page. Les résultats de l'évaluation ont montré que les jeux de données générés pour trois types de DSSs (fiches de paie, tickets de caisse et factures) présentent un degré élevé de diversité, ce qui permet d'éviter le sur-apprentissage lors de l'entraînement des systèmes d'extraction d'informations. En s'appuyant sur le format spécifique des DSSs, constitué de paires de mots (mots-clés, informations) situés dans des voisinages proches spatialement, le document est modélisé sous forme de graphe où les nœuds représentent les mots et les arcs, les relations de voisinage. Le graphe est incorporé dans un réseau d'attention à graphe (GAT) multi-couches (Multi-GAT). Celui-ci applique le mécanisme d'attention multi-têtes permettant d'apprendre l'importance des voisins de chaque mot pour mieux le classer. Une première version de ce modèle a été utilisée en mode supervisé et a obtenu un score F1 de 96 % sur deux jeux de données de factures et de fiches de paie générées, et de 89 % sur un ensemble de tickets de caisse réels (SROIE). Nous avons ensuite enrichi le Multi-GAT avec un plongement multimodal de l'information au niveau des mots (avec des composantes textuelle, visuelle et positionnelle), et l'avons associé à un auto-encodeur variationnel à graphe (VGAE). Ce modèle fonctionne en mode semi-supervisé, capable d'apprendre à partir des données annotées et non annotées simultanément. Pour optimiser au mieux la classification des nœuds du graphe, nous avons proposé un semi-VGAE dont l'encodeur partage ses premières couches avec le classifieur Multi-GAT. Cette optimisation est encore renforcée par la proposition d'une fonction de perte VGAE gérée par la perte de classification. En utilisant une petite base de données non annotées, nous avons pu améliorer de plus de 3 % le score F1 obtenu sur un ensemble de factures générées. Destiné à fonctionner dans un environnement protégé, nous avons adapté l'architecture du modèle pour son chiffrement homomorphe. Nous avons étudié une méthode de réduction de la dimensionnalité du modèle Multi-GAT. Ensuite, nous avons proposé une approche d'approximation polynomiale des fonctions non-linéaires dans le modèle. Pour réduire la dimension du modèle, nous avons proposé une méthode de fusion de caractéristiques multimodales qui nécessite peu de paramètres supplémentaires et qui réduit les dimensions du modèle tout en améliorant ses performances. Pour l'adaptation au chiffrement, nous avons étudié des approximations polynomiales de degrés faibles aux fonctions non-linéaires avec une utilisation des techniques de distillation de connaissance et de fine tuning pour mieux adapter le modèle aux nouvelles approximations. Nous avons pu minimiser la perte lors de l'approximation d'environ 3 % pour deux jeux de données de factures ainsi qu'un jeu de données de fiches de paie et de 5 % pour SROIE.

Mots-clés: Extraction d'informations, document semi-structuré (DSS), apprentissage semi-supervisé,

réseau d'attention à graphe (GAT), auto-encodeur variationnel à graphe (VGAE), chiffrement homomorphe.

Abstract

This thesis is being carried out as part of the BPI DeepTech project, in collaboration with the company Fair&Smart, primarily looking after the protection of personal data in accordance with the General Data Protection Regulation (RGPD). In this context, we have proposed a deep neural model for extracting information in semi-structured administrative documents (SSDs). Due to the lack of public training datasets, we have proposed an artificial generator of SSDs that can generate several classes of documents with a wide variation in content and layout. Documents are generated using random variables to manage content and layout, while respecting constraints aimed at ensuring their similarity to real documents. Metrics were introduced to evaluate the content and layout diversity of the generated SSDs. The results of the evaluation have shown that the generated datasets for three SSD types (payslips, receipts and invoices) present a high diversity level, thus avoiding overfitting when training the information extraction systems. Based on the specific format of SSDs, consisting specifically of word pairs (keywords-information) located in spatially close neighborhoods, the document is modeled as a graph where nodes represent words and edges, neighborhood connections. The graph is fed into a multi-layer graph attention network (Multi-GAT). The latter applies the multi-head attention mechanism to learn the importance of each word's neighbors in order to better classify it. A first version of this model was used in supervised mode and obtained an F1 score of 96% on two generated invoice and payslip datasets, and 89% on a real receipt dataset (SROIE). We then enriched the Multi-GAT with multimodal embedding of word-level information (textual, visual and positional), and combined it with a variational graph auto-encoder (VGAE). This model operates in semi-supervised mode, being able to learn on both labeled and unlabeled data simultaneously. To further optimize the graph node classification, we have proposed a semi-VGAE whose encoder shares its first layers with the Multi-GAT classifier. This is also reinforced by the proposal of a VGAE loss function managed by the classification loss. Using a small unlabeled dataset, we were able to improve the F1 score obtained on a generated invoice dataset by over 3%. Intended to operate in a protected environment, we have adapted the architecture of the model to suit its homomorphic encryption. We studied a method of dimensionality reduction of the Multi-GAT model. We then proposed a polynomial approximation approach for the non-linear functions in the model. To reduce the dimensionality of the model, we proposed a multimodal feature fusion method that requires few additional parameters and reduces the dimensions of the model while improving its performance. For the encryption adaptation, we studied low-degree polynomial approximations of nonlinear functions, using knowledge distillation and fine-tuning techniques to better adapt the model to the new approximations. We were able to minimize the approximation loss by around 3% on two invoice datasets as well as one payslip dataset and by 5% on SROIE.

Keywords: Information extraction, semi-structured document (SSD), semi-supervised learning, Graph Attention Network (GAT), Variational Graph Auto-Encoder (VGAE), homomorphic encryption.

Table des matières

Chapitre 3
Système semi-supervisé pour l'extraction d'informations dans les DSSs

Table des figures

Liste des tableaux

Liste des abréviations

ACP	Analyse en Composantes Principales
BGV	Brakerski-Gentry-Vaikuntanathan
BiGRU	Bidirectional Gated Recurrent Unit
BLSTM	Bidirectional Long Short-Term Memory
BPE	Byte-Pair-Encoding
CBOW	Continuous Bag Of Words
CKKS	Cheon-Kim-Kim-Song
CNN	Convolutional Neural Network
PCA	Principal Component Analysis
CRF	Conditional Random Fields
DSS	Document semi-structuré
EI	Extraction d'Informations
FPN	Feature Pyramid Network
GAN	Generative adversial Network
GAT	Graph Attention Network
GCN	Graph Convolutional Network
GCNNs	Graph Convolutional Neural Networks
GGNN	Gated Graph Neural Networks
GloVE	Global Vectors for Word Representation
GPT	Generative Pre-Training
HE	Homomorphic Encryption
IOB	Inside, Outside, Beginning
KL-Divergence	Kullabck-Leibler Divergence
KPCA	Kernel-based Principal Component Analysis
LDA	Linear Discriminant Analysis
LSTM	Long short-term memory
MLP	Multilayer perceptron
MPC	Secure Multi-Party Computation
NLTK	Natural language Tool Kit
OCR	Optical Character Recognition
OOV	Out-Of-Vocabulary

ReLU	Rectified Linear Unit
RNN	Recurrent Neural Network
RoIAlign	Region of Interest Align
SELF-BLEU	SELF-BiLingual Evaluation Understudy
TAL	Traitement automatique de la langue
VAE	Variational Auto-Encoder
VGAE	Variational Graph Auto-encoder
XML	eXtensible Markup Language

1

Introduction

Depuis la mise en vigueur du RGPD (Règlement Général sur la Protection des Données) le 25 mai 2018, les citoyens européens bénéficient d'une base légale leur permettant de contrôler et de récupérer leurs données personnelles. Ce dispositif vise à fournir des moyens techniques simples et accessibles au plus grand nombre de personnes pour administrer leurs données et les partager, tout en permettant aux organismes qui les utilisent de se conformer à la législation en vigueur. Une plate-forme sécurisée pour la gestion automatique des documents des utilisateurs est proposée dans le cadre du projet BPI DeepTech, avec notre équipe, en collaboration avec les sociétés Fair&Smart et CryptoExperts et l'équipe COAST de l'INRIA. Elle offre un système d'extraction d'informations (EI) chiffrées tout en garantissant la confiance entre les utilisateurs de la plate-forme.

L'EI dans les documents administratifs tels que les factures et les fiches de paie, se révèle être une tâche importante dans la gestion des documents dans les entreprises, pour la gestion du courrier par exemple ou le traitement de l'information, de manière générale. Ces documents administratifs, dits documents semi-structurés (DSSs), sont le plus souvent présentés dans un format très spécifique. Le traitement automatique de ce type de documents continue de poser un vrai défi en recherche, en raison de la variété de leur contenu et de leur mise en page. L'équipe READ, en collaboration avec Fair&Smart, avait proposé, dans un précédent travail, un premier système de réseaux de neurones profonds pour l'extraction d'informations à partir de documents de type facture. La facture est représentée par un graphe de mots, où chaque mot est connecté à ses quatre voisins les plus proches dans les quatre directions cardinales. Le système utilise un réseau convolutif à graphe (GCN). Il applique une convolution spectrale avec des filtres de Chebyshev au graphe modélisant la facture. Ce GCN s'est montré capable d'extraire 27 entités avec un score F1 de 93% à partir d'un ensemble de factures artificielles, en apprenant à prédire les classes des nœuds (mots) dans les graphes des documents.

Dans cette thèse, nous étendons ce système à d'autres types de documents administratifs semi-structurés tels que les fiches de paie et les tickets de caisse. L'objectif est de proposer un modèle neuronal profond adapté à tous les DSSs, en prenant en compte le manque de données d'entraînement étiquetées et la nécessité de sécuriser le modèle et les données. Cette dernière contrainte va influer sur la méthode de chiffrement du système, tâche réalisée par un autre partenaire du projet spécialisé dans la sécurité des produits, la société CryptoExperts, mais où nous nous préoccuperons de la simplification du modèle et de son adaptation au chiffrement.

Il existe dans la littérature plusieurs systèmes d'EI dans les documents administratifs qui s'appuient sur des approches d'apprentissage profond supervisé tels que ceux proposés par [25, 79, 94, 96, 129, 139, 145]. Ces systèmes proposent différentes modélisations de documents soit à base

de séquences de texte, soit à base de graphes ou de grilles. Ils peuvent prendre en considération plusieurs types de caractéristiques, notamment les caractéristiques textuelles, visuelles et/ou positionnelles. Les principaux problèmes auxquels sont confrontés ces systèmes sont la complexité et le coût de la tâche d'étiquetage des documents, ainsi que le manque de jeux de données d'entraînement réels. Pour pallier cela, d'autres méthodes ont été proposées, comme [36, 41, 93, 135] qui effectuent un transfert d'apprentissage en ajoutant une étape de pré-entraînement sur des données non étiquetées, provenant de domaines externes (textes libres/documents ordinaires). Ils pré-entraînent un modèle en utilisant différentes tâches telles que la reconstruction de l'entrée, la prédiction de la phrase suivante, etc., en mode non supervisé, et affinent ensuite le résultat sur des documents étiquetés. Toutefois, ces méthodes nécessitent d'énormes bases de données d'entraînement pour l'étape du pré-entraînement et forment des modèles complexes (des centaines de millions de paramètres).

Dans le cadre de cette thèse, nous avons conçu un système semi-supervisé moins complexe que les systèmes de pointe d'EI dans les DSSs, qui apprend simultanément à partir de documents étiquetés et non étiquetés du même domaine. Ce système est basé sur un classifieur de nœuds de graphes Multi-GAT et un auto-encodeur variationnel à graphe (VGAE). En analysant les DSSs, nous avons constaté qu'ils contiennent des classes d'informations qui sont physiquement regroupées et identifiées par quelques mots-clés indicatifs dans leur voisinage. Ce dernier est très important ; c'est le cas, par exemple, pour les factures, des adresses, des informations de l'entreprise et des identifiants des employés. Nous avons donc choisi de modéliser le DSS par un graphe de mots qui prend en compte ces spécificités. Pour exploiter au mieux la disponibilité de mots-clés dans le voisinage des informations, nous avons proposé une méthode efficace de sélection des voisins les plus proches susceptibles d'être les plus indicatifs. En outre, nous avons fusionné plusieurs caractéristiques multimodales des mots, de type textuel, positionnel et visuel. Le classifieur que nous avons choisi est basé sur un GAT (Graph Attention Network) qui apprend l'importance des mots voisins dans le graphe et aide à prédire les classes/entités des mots. Pour maximiser le processus d'optimisation de la classification sur les données étiquetées et non étiquetées, nous avons proposé une architecture semi-supervisée adaptée (le modèle Multi-GAT+VGAE) qui est une association du Multi-GAT et du VGAE. Elle se distingue par un classifieur et un encodeur qui partagent leurs premières couches GAT et une fonction objective gérée par la perte de classification. Cela a permis au modèle d'être orienté le plus efficacement possible vers une optimisation maximale de la classification.

Nous avons également traité dans cette thèse le problème de la sécurisation des données et de leur traitement. L'évaluation des modèles d'EI à l'étape de l'inférence dans des environnements confidentiels est une tâche cruciale pour un grand nombre d'industries. Lors de l'extraction d'informations à partir de documents confidentiels, il est nécessaire de chiffrer les données avant de les charger sur le serveur. Il est également important de s'assurer que le modèle d'inférence peut évaluer les données chiffrées avec précision. Le chiffrement homomorphe (HE : Homomorphic Encryption) permet d'effectuer des calculs directement sur les données chiffrées, mais ne fournit qu'un nombre limité d'opérations arithmétiques et de fonctions. La plupart des systèmes performants d'extraction d'informations à partir de documents administratifs ont des architectures complexes avec un grand nombre de paramètres entraînables, ce qui rend leur chiffrement plus coûteux. De plus, ils ne sont pas compatibles avec les protocoles de chiffrement existants en raison de la nature des fonctions non-linéaires utilisées. L'utilisation de ces systèmes sur des plateformes sécurisées et chiffrées nécessite leur adaptation aux protocoles de chiffrement existants.

Pour relever les principaux défis du protocole HE, nous avons proposé une approche permettant d'adapter le modèle d'inférence Multi-GAT au chiffrement homomorphe. Cette approche réduit la dimensionnalité des données et du modèle et fournit une approximation polynomiale à

toutes les opérations non-linéaires dans le Multi-GAT, tout en minimisant la perte de précision, le tout ne nécessitant pas de connexions multiples entre le client et le serveur.

Le reste de la thèse est structuré en cinq chapitres. Dans le chapitre 2, nous présentons une étude bibliographique des différentes représentations existantes des modalités de texte dans les documents, à savoir le plongement du texte, la mise en page et le plongement de l'image ; les différentes méthodes d'extraction d'informations et également les méthodes d'apprentissage dans les réseaux de convolution à graphe.

Le chapitre 3 présente en premier notre approche de modélisation et de génération des documents semi-structurés. Nous détaillons la méthode générique développée pour la génération automatique de données d'entraînement de type DSS ainsi que les différentes métriques proposées pour évaluer leur diversité. Ce chapitre expose également en détail les différentes versions de notre modèle d'EI dans les DSSs. La deuxième section décrit la première version du modèle supervisé transductif $Multi-GAT_{Dataset}$. La troisième section présente les améliorations apportées à la première version donnant lieu à un modèle multimodal inductif $Multi-GAT_{DSS}$. Enfin, la dernière section est consacrée à l'extension du modèle au mode semi-supervisé en incorporant un auto-encodeur variationnel à graphe qui permet d'apprendre à partir de documents non annotés.

Dans le chapitre 4, nous étudions les méthodes de réduction de la dimensionnalité des données et d'adaptation du modèle d'inférence au chiffrement homomorphe. La première section est consacrée à la réduction de la dimensionnalité des données et du modèle. Nous commençons par présenter un état de l'art des méthodes de réduction de dimensionnalité par sélection et extraction de caractéristiques. Ensuite, nous détaillons notre approche choisie pour réduire la dimension de nos données tout en préservant les performances du modèle d'inférence. La deuxième section est consacrée à la méthode d'adaptation du modèle d'inférence au protocole de chiffrement homomorphe. Cette section expose en premier les différentes méthodes existantes dans la littérature pour approximer les opérations non-linéaires dans les réseaux de neurones. Ensuite, nous expliquons l'approche adoptée pour l'approximation polynomiale des opérations non-linéaires dans le modèle d'inférence Multi-GAT.

Nous concluons cette thèse dans le chapitre 5 par une discussion des travaux et contributions accomplies et présentons quelques perspectives.

2

État de l'art sur l'extraction d'informations

Sommaire

2.1 Introduction

L'extraction d'informations (EI) dans les documents s'effectue le plus souvent selon le pipeline illustré dans la Figure 2.1.

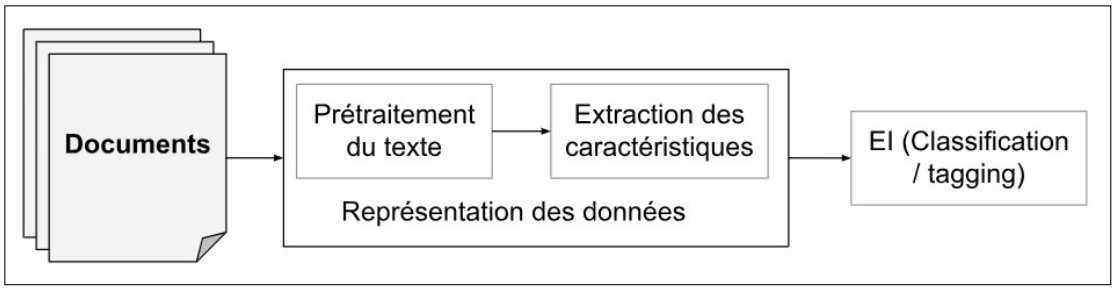


FIGURE 2.1 – Le pipeline de l'extraction d'informations dans les documents.

Une première étape cruciale de la tâche d'EI consiste à représenter le contenu du document dans un format compréhensible par les modèles neuronaux. Cette étape comprend deux parties

essentielles : le prétraitement du texte et l'extraction des caractéristiques. Ensuite, il s'agit d'assurer la classification ou l'étiquetage des représentations apprises à l'aide d'un modèle neuronal adéquat et pertinent.

2.2 Prétraitement

L'étape de prétraitement consiste à détecter et à extraire les segments de texte et à fournir des informations sur les boîtes englobantes qui les entourent (position, dimensions, etc.) dans le document. Cette étape est le plus souvent réalisée à l'aide d'un outil de reconnaissance optique de caractères (OCR) ou d'un composant intégré dans l'architecture neuronale globale, formant ainsi une architecture d'EI de bout en bout. D'autres techniques de prétraitement du texte, telles que la segmentation ou la tokenisation, la conversion en minuscules, la suppression des mots vides et la lemmatisation [105], peuvent également être appliquées à ce stade.

La technique de tokenisation, par exemple, correspond au processus de séparation du texte (phrases) en mots, symboles, caractères, tokens ou sous-mots [6]. Généralement, la tokenisation est la première étape de prétraitement standard dans toute tâche de Traitement Automatique de la Langue (TAL) [44].

La conversion de tous les mots en minuscules [37] lors de la phase de prétraitement a pour effet de fixer la représentation des mots identiques, sans faire de distinction entre leur écriture en majuscules et en minuscules. Bien que l'utilisation de majuscules dans un corpus puisse perturber la classification du texte dès lors qu'un même mot peut avoir différentes représentations, certains mots peuvent encore poser des problèmes d'interprétation. Par exemple, le mot « US » en majuscule peut être utilisé à la fois comme pronom et comme abréviation d'un nom de pays en anglais, ce qui rend la conversion en minuscules, problématique [105].

Les mots vides peuvent également être supprimés du texte, en particulier dans les documents non structurés contenant du texte organisé en paragraphes. Les mots vides comprennent les conjonctions telles que « ou », « et » et « mais », les pronoms tels que « il », « elle » et « ils », etc. [70]. Ils peuvent être supprimés pour améliorer la clarté du texte. La suppression de ces mots n'a habituellement pas de conséquence sur le sens général du texte à classer ou à extraire. En outre, les caractères spéciaux ou les chiffres inutiles peuvent également être supprimés s'ils n'ajoutent pas de valeur significative au processus de l'EI [70].

L'étape de prétraitement peut également inclure la lemmatisation ou le stemming qui consiste à remplacer et à supprimer les suffixes et les préfixes pour obtenir la racine ou le mot de base, étant donné qu'un mot peut prendre de nombreuses formes différentes, mais que leur signification sémantique reste identique. Les caractéristiques des mots peuvent donc être réduites grâce à cette technique de lemmatisation [100].

2.3 Extraction de caractéristiques

L'extraction de caractéristiques est le processus de conversion de données brutes en données numériques interprétables par la machine. En analyse neuronale, cette étape est réalisée par vectorisation des caractéristiques. Les plus pertinentes à extraire sont les caractéristiques textuelles. Cependant, d'autres peuvent s'y ajouter pour les enrichir comme la position et l'aspect visuel.

Nous présenterons brièvement dans la section suivante les algorithmes d'extraction de ces trois types de caractéristiques les plus répandus.

2.3.1 Caractéristiques textuelles

Le calcul des caractéristiques textuelles, connu également sous le nom de « plongement de mots » (*word embedding*), est une technique de traitement du langage naturel qui présente le texte d'un corpus sous forme de vecteurs. Il existe deux catégories de représentation de mots, selon la prise en compte ou non du contexte : les représentations non contextuelles et les représentations contextuelles. Dans la première catégorie, la représentation du texte obtenue après l'apprentissage du modèle est fixe et ne dépend pas du contexte de l'entrée. En revanche, dans la seconde catégorie, chaque mot se voit attribuer une représentation qui change suivant la phrase d'entrée, de sorte que la représentation des mots varie en fonction des différents contextes linguistiques.

Les algorithmes les plus répandus de chaque catégorie de représentation sont décrits ci-après.

Représentation non contextuelle des mots

Les modèles de représentation non contextuelle de mots, telles que GloVe [98], word2vec [102], et fastText [12], encodent chaque mot, appartenant à un vocabulaire, sous forme d'un vecteur qui capture ses informations sémantiques, comme l'analogie et la similarité entre les mots. Cela permet de s'assurer que les vecteurs de représentation de mots similaires tels que les synonymes, restent similaires. Pour obtenir une meilleure représentation, ces vecteurs sont pré-entraînés sur de vastes corpus linguistiques, à l'aide de diverses techniques. Après l'entraînement, ces modèles conservent les représentations des mots non modifiées en fonction de leur contexte c.à.d. n'ayant subi aucune influence de leur contexte.

L'un des premiers modèles proposés dans cette catégorie est Word2vec [102]. Il utilise des modèles de réseaux neuronaux CBOW ou Skip-gram [101] pour prédire les représentations de mots en fonction de leur contexte local. Ce modèle acquiert des connaissances sémantiques en analysant une fenêtre de contexte local dans le corpus d'apprentissage, ce qui lui permet de prédire les mots en fonction de ce contexte. Cependant, Word2vec n'utilise pas efficacement les informations statistiques globales du corpus, comme la distribution des mots dans la langue ou le corpus.

Pour pallier cette limite, le modèle GloVe [98] a été introduit. GloVe est un algorithme de calcul de matrice de co-occurrence globale où chaque élément représente la fréquence de co-occurrence entre les mots de la ligne et de la colonne correspondantes dans une fenêtre donnée. L'algorithme comporte deux étapes : la première consiste à créer la matrice de co-occurrence à partir du corpus et la seconde, à la factoriser pour obtenir des vecteurs. GloVe utilise des fenêtres de contenu local et la factorisation de la matrice pour produire une statistique de co-occurrence globale et cumulative de mots. Il fournit également un ensemble de vecteurs pré-entraînés dans plusieurs dimensions (100, 200, 300). Cependant, Word2vec [102] et GloVe [98] ne peuvent pas générer des plongements de mots hors vocabulaire (OOV : Out Of Vocabulary). Au lieu de cela, ils attribuent un vecteur aléatoire à tout mot qu'ils n'ont pas rencontré au cours de l'apprentissage.

D'autres modèles ont été proposés pour résoudre ce problème. Bojanowski et al. [12] ont proposé le modèle fastText qui peut être considéré comme une extension du modèle Word2vec, puisqu'il est lui aussi basé sur les algorithmes CBOW et skip-gram. FastText décompose chaque mot en n-grammes de caractères pour alimenter le réseau neuronal qui apprend la relation entre les caractères et capture la sémantique des mots. Il offre de meilleurs résultats que les modèles décrits précédemment grâce à une meilleure représentation des mots, en particulier dans le cas des mots rares. Il gère également le cas des mots hors vocabulaire, contrairement à Word2Vec et GloVe. Enfin, il fournit des modèles pré-entraînés pour 294 langues différentes, avec des vecteurs de dimension 300, entraînés sur Wikipédia.

Enfin, une dernière technique de représentation non-contextuelle est BPE (Byte-Pair Encoding) [122]. C'est une technique de segmentation non supervisée qui implique la fusion itérative des symboles adjacents les plus fréquents en de nouveaux symboles. L'ensemble des symboles acquis de cette manière est identifié comme le vocabulaire BPE. L'exécution de BPE, c'est-à-dire l'exécution répétitive des opérations de fusion apprises, permet de segmenter le texte en sous-mots, ce qui permet d'éviter efficacement le problème de l'OOV. BPEmb [54] est une collection de modèles pré-entraînés d'embedding de sous-mots qui implémentent l'algorithme BPE dans 275 langues. Ils sont entraînés en utilisant trois types d'architecture : la moyenne, un CNN et un RNN. Cette dernière architecture a enregistré les meilleurs résultats sur la tâche de la classification fine des types d'entités dans le texte de Wikipedia. Différentes tailles de vecteurs d'embedding sont proposées, en occurrence 25, 50, 100, 200 et 300, ainsi que plusieurs tailles de vocabulaire.

Représentation contextuelle des mots

La représentation contextuelle des mots vise à encoder le sens ou la fonction d'un mot dans un contexte particulier en tenant compte des mots voisins. Son principal avantage par rapport à la représentation non contextuelle est la capacité de distinguer la sémantique de deux mots identiques dans des contextes différents. Par exemple, Peters et al. [112] proposent le modèle ELMo (Embedding from Language Models) qui génère des représentations de mots par le biais d'un réseau BLSTM. Ce réseau encode le contexte du mot en considérant ses voisins de gauche et de droite. Par ailleurs, [36, 93, 116] obtiennent les représentations à l'aide de modèles basés sur l'architecture du Transformer [125]. Tous ces modèles peuvent offrir des représentations distinctes du même mot dans des contextes différents. Dans ce qui suit, notre attention se porte sur l'ensemble des modèles utilisant les Transformers [125], car ils sont les plus répandus dans cette catégorie et ils se sont également avérés plus efficaces et plus rapides que les architectures LSTM [56] pour la modélisation du langage.

Le modèle linguistique pré-entraîné basé sur le Transformer GPT (Generative Pre-Training), proposé par [116], représente le premier modèle de langage de ce type avec la capacité de manipuler efficacement le sens des mots, en fonction du contexte. GPT est entraîné de manière non supervisée sur un vaste corpus de textes libres. Néanmoins, le modèle présente l'inconvénient de posséder une nature unidirectionnelle : il n'a été entraîné qu'à prédire le contexte futur à droite du texte. Pour surmonter cet inconvénient, un modèle linguistique plus avancé appelé BERT (Bidirectional Encoder Representations from Transformers) [36] a été proposé. BERT est un modèle bidirectionnel, ce qui signifie qu'il prend en compte le contexte à droite et à gauche du texte. Dans son entraînement, BERT utilise deux tâches d'apprentissage non supervisé, y compris un modèle de langage masqué, où des tokens aléatoires sont masqués et prédits, et une tâche visant à prédire la phrase suivante. BERT a été entraîné sur la base de données du corpus Books [155] et sur des passages du texte anglais de Wikipedia.

Une autre approche hautement optimisée pour BERT a ensuite été développée, connue sous le nom de RoBERTa (Robustly Optimized BERT Pretraining Approach) [93]. RoBERTa est une version simplifiée de BERT avec moins de paramètres et de meilleures performances. Les modifications apportées au modèle BERT comprennent un entraînement plus long avec des lots plus importants de données, l'élimination de l'entraînement sur la tâche de prédiction de la phrase suivante, et un entraînement sur des séquences plus longues. Toutes ces modifications ont toutefois permis d'améliorer les performances du modèle.

Bien que ces modèles puissent résoudre les problèmes contextuels, ils nécessitent d'énormes quantités de données et plus de centaines de millions de paramètres pour leur entraînement. En

outre, pour traiter les mots hors vocabulaire qui n'ont pas été rencontrés au cours de l'apprentissage, ils leur attribuent le token « UNK » et le même vecteur, ce qui donne des résultats non optimaux s'il y a un nombre important de mots hors vocabulaire.

D'autres modalités telles que celles liées à la mise en page du document et à l'image, devraient également être prises en compte pour enrichir le vecteur de caractéristiques des mots.

2.3.2 Caractéristiques positionnelles

La position 2D illustre la relation spatiale entre les éléments de texte dans un document et peut donc fournir davantage de caractéristiques utiles à la classification du texte dans le document.

La page du document est habituellement considérée comme un référentiel de coordonnées dont l'origine se situe en haut à gauche. La boîte englobante du texte et son vecteur de plongement peuvent alors être définis de diverses manières. La boîte englobante dans LayoutLM [135] et TRIE [145] par exemple, est définie précisément par (x_0, y_0, x_1, y_1) où (x_0, y_0) correspond à la position du coin supérieur gauche de la boîte englobante et (x_1, y_1) à la position du coin inférieur droit. Quatre couches de plongements positionnels sont ajoutées avec deux tables de plongements, où les positions x_0 et x_1 sont extraites de la table de plongements X et les positions y_0 et y_1 sont extraites de la table de plongements Y. En revanche, TRIE [145] utilise une couche de plongement sur les quatre coordonnées des boites englobantes. Quant à LayoutLMv2 [134] et StructText [84], ils représentent la boîte englobante à l'aide de six éléments $(x_0, y_0, x_1, y_1, largeur, hauteur)$. Comme LayoutLM, ils normalisent et discrétisent toutes ces coordonnées en nombres entiers dans l'intervalle $[0, 1000]$ et utilisent deux couches de plongement partagées. Dans LayoutLMv2, la couche de plongement de la position concatène les six caractéristiques de la boîte englobante pour construire une représentation de la position en 2D au niveau du token. StructText [84] utilise aussi une couche de plongement pour encoder chaque segment ou mot. Par ailleurs, LAMBERT [41] commence par normaliser les boîtes englobantes en divisant leurs dimensions par la hauteur de la page, ce qui les transforme de sorte que le coin supérieur gauche se trouve à la position $(0, 0)$ et que la boîte englobante de la page devienne $(0, 0, w, 1)$, où w est la largeur normalisée. Le plongement en 2D d'un élément est alors défini comme la concaténation des quatre représentations de coordonnées de sa boîte englobante. Ils définissent le plongement correspondant à une seule coordonnée et utilisent deux encodages de plongement sinusoïdaux basés sur les fonctions sinus (sin) et cosinus (cos). En effet, les fonctions sinus et cosinus ont des valeurs dans [-1, 1], ce qui maintient les valeurs de la matrice d'encodage de la position dans une plage normalisée. La combinaison de ces fonctions sinusoïdales permet de capturer les relations entre les positions. En d'autres termes, elle peut saisir le fait que la position i d'un élément est plus proche de la position $i + 1$ que de la position $i + 9$, par exemple.

2.3.3 Caractéristiques visuelles

La représentation de l'image sert à saisir certaines caractéristiques visuelles au niveau des mots ou des segments du texte telles que le type, la couleur des polices de caractères et les styles tels que le gras, le souligné et l'italique. En combinant les caractéristiques des images avec les représentations textuelles traditionnelles, on obtient donc des représentations plus riches

Avec la boîte englobante de chaque mot, provenant des résultats de l'OCR, LayoutLM [135] segmente l'image en plusieurs éléments qui correspondent aux mots. Les caractéristiques visuelles de ces éléments d'image sont extraites par le modèle Faster R-CNN [118]. LayoutLMv2 [134] quant à lui utilise quatre éléments (tokens) visuels du document, séparés des éléments textuels.

Il calcule les caractéristiques visuelles de la page à l'aide d'un encodeur basé sur l'architecture ResNeXt-FPN [87, 133] qui convertit l'image du document en une carte de caractéristiques. Celle-ci est ensuite aplatie en une séquence de plongements visuels. Une couche de projection linéaire est ensuite appliquée à chaque token visuel afin d'unifier leur dimensionnalité avec les tokens textuels.

PICK [140] utilise le modèle ResNet [53] comme base pour obtenir les caractéristiques visuelles et extraire les informations morphologiques des segments de texte dans l'image du document. Cette représentation visuelle est ensuite combinée avec la représentation textuelle de chaque segment à l'aide d'une opération d'addition de caractéristiques. Par ailleurs, TRIE [145] adopte le Feature Pyramid Network (FPN) [87] avec ResNet [53] comme encodeur CNN pour l'extraction des caractéristiques visuelles de l'image. Compte tenu des positions des segments du texte, RoIAlign [52] est ensuite appliqué sur la sortie du CNN afin d'extraire les caractéristiques de chaque segment. Enfin, chaque vecteur du contexte visuel est combiné avec le vecteur du contexte textuel correspondant en utilisant une somme pondérée afin d'obtenir le vecteur final du segment. StructText [84] utilise également ResNet50 [133] avec FPN [87] suivie du RoIAlign [52] pour extraire les caractéristiques visuelles de chaque segment de texte.

2.4 Modèles d'EI

Les systèmes d'EI peuvent être regroupés en plusieurs catégories en fonction du type de modélisation des documents et du mode d'apprentissage.

2.4.1 Classement par type de modélisation

Selon les méthodes de modélisation des documents adoptées, on peut distinguer trois types d'approches : les approches basées sur les séquences de segments de texte, les approches basées sur les grilles et les approches basées sur les graphes.

Séquences de segments de texte

De nombreux articles de la littérature ont abordé la tâche du traitement des documents comme un problème d'étiquetage de séquences et ont utilisé des modèles de réseaux neuronaux récurrents pour extraire les entités nommées comme des noms de personnes, des organisations et des lieux. Les auteurs de [79, 92, 96, 140, 145] ont tous utilisé un BLSTM suivi d'une couche CRF [78] pour traiter la tâche d'étiquetage de séquences de texte. Ce type de réseau neuronal (BLSTM) s'est avéré efficace pour les problèmes de séquences. En effet l'ordre des mots dans les séquences du texte est important et les BLSTM sont capables d'apprendre des dépendances à long terme et de se souvenir de longues séquences d'entrée car elles disposent d'une mémoire interne pour toutes les observations en entrée. La couche CRF [78] peut utiliser efficacement les tags passés et futurs pour prédire le tag courant dans une phrase. Dans ces travaux, le document est considéré comme une séquence de segments de texte (caractéristiques) et les entités de la séquence sont étiquetées à l'aide du format IOB (abréviation de Inside, Outside, Beginning). Ce qui différencie chaque approche est le choix du calcul des caractéristiques des segments de texte qui représentent l'entrée du BLSTM.

Ma et al. [96] utilisent un CNN pour calculer la représentation des mots au niveau des caractères. Le vecteur de représentation des caractères est ensuite concaténé avec le plongement des mots avant d'être introduit dans le réseau BLSTM. Liu et al. [92] concatènent le plongement du graphe avec le plongement textuel du token et les introduisent dans le BLSTM-CRF

standard. Ils utilisent des vecteurs Word2Vec pour le plongement des tokens et une architecture de graphe convolutionnel pour produire un plongement de graphe. Cela permet d'obtenir un contexte visuellement riche et d'ajouter des informations contextuelles à la séquence d'entrée. TRIE [145] utilise des informations textuelles et visuelles (couleur, disposition, etc.) ainsi que la position comme entrée du BLSTM. Cette approche intègre la reconnaissance de texte et l'extraction d'informations dans un système de bout en bout. PICK [140] combine le plongement du texte et le plongement de l'image et ajoute le résultat d'une convolution à graphe, puis introduit cette combinaison dans le réseau neuronal récurrent. La convolution à graphe vise à exploiter le voisinage des segments de texte par l'apprentissage d'une matrice adjacente souple. Il a déjà été démontré que ce type d'architecture a une capacité limitée à apprendre le lien entre des mots éloignés.

Grille 2D

Les approches basées sur les grilles telles que celles de Chargrid [71], de Dang et al. [28], de CUTIE [149], ou de BERTgrid [35] considèrent chaque document comme une grille 2D de plongements de tokens et utilisent des modèles de segmentation standard pour extraire les valeurs des champs dans la grille 2D.

Chargrid [71] présente une grille de caractères construite à partir des boîtes englobantes des caractères dans l'image du document. L'ensemble du texte dans le document est représenté comme une collection de tuples comprenant les caractères et les caractéristiques de leurs boîtes englobantes correspondantes (coordonnées et dimensions). Ces tuples sont ensuite utilisés pour construire la grille de caractères de la page originale, où chaque zone couverte par un caractère se voit attribuer une valeur constante et tous les autres pixels correspondant à des zones vides de la page originale sont initialisés à 0. Cette représentation de la grille de caractères est transmise à un CNN qui effectue une segmentation sémantique et prédit une classe pour chaque pixel de caractère dans le document. Dang et al. [28] effectue également une tâche de segmentation sémantique au niveau des pixels pour étiqueter et extraire des informations pertinentes à partir de la grille de caractères comme Chargrid [71]. Cependant, ils s'appuient sur une architecture d'encodeur-décodeur basée sur le modèle U-Net couplé [28] et ils exploitent également le mécanisme d'auto-attention [143] et la couche de convolution en boîte [14] pour optimiser leurs résultats.

Au lieu de construire une grille au niveau du caractère comme le proposent Katti et al. [71], BERTgrid [35] construit une grille au niveau des mots et incorpore des vecteurs contextuels des mots provenant du modèle de langage BERT [36]. L'image du document est tout d'abord introduite à un OCR pour récupérer les mots et leurs positions qui sont ensuite transmis au modèle BERT afin d'obtenir le vecteur contextuel de chaque mot. A partir des informations des positions, BERTgrid construit la grille 2D comme dans Chargrid [71] et utilise également la même architecture du réseau CNN pour extraire les segments à partir du document. VisualWordGrid [72], à son tour, étend l'approche de Chargrid en ajoutant l'aspect visuel aux aspects textuels et de mise en page du document. Au lieu d'encoder le document au niveau du caractère, VisualWordGrid encode le document au niveau du mot en utilisant les plongements Word2vec [102] ou fastText [12]. Il utilise Unet [119] comme modèle de segmentation et ResNet34 [53] comme base pour l'encodeur afin de classer les entités.

Cependant, ces approches, basées sur des grilles, se sont avérées moins performantes que d'autres méthodes telles que LayoutLM [135], LAMBERT [41] et TRIE [145].

Graphe

D'autres approches modélisent le document sous forme de graphe dont les nœuds représentent des segments de texte (mots, ensembles de mots ou lignes de texte).

Certaines simplifient le processus d'analyse du document en classant les nœuds dans des graphes tout en établissant un type de voisinage pour garantir une classification correcte des nœuds. Lohani et al. [94] représentent les factures sous forme de graphes où les nœuds correspondent aux mots du document, et chaque nœud est lié à ses quatre voisins les plus proches suivant les quatre points cardinaux. Cette exploitation du voisinage local est réalisée via un modèle de réseau convolutif à graphe (GCN). Une implémentation de la convolution à l'aide de polynômes de Chebyshev, permettant une classification rapide des nœuds, est proposée. Hua et al. [57] modélisent également un document avec une structure de graphe orienté. Chaque nœud représente un segment de texte (séquence de texte) et est connecté à tous les autres segments de texte appartenant à la même ligne et à la ligne du dessus. Un nœud global est ajouté pour relier tous les nœuds locaux. Enfin, un réseau neuronal à graphe basé sur le mécanisme d'attention avec une couche CRF est appliqué pour étiqueter les séquences de texte. Rusinol et al. [120], quand à eux, utilisent des graphes étoiles pour extraire les champs d'une facture. Dans ce graphe, les nœuds représentent les mots du document et les arêtes correspondent aux relations spatiales entre chaque champ et les autres mots. Après une première phase d'apprentissage, les auteurs utilisent un système de vote pour déterminer la position du champ le plus probable à extraire, sur la base de modèles appris précédemment (graphes étoiles), stockés dans une base de données. Hammami et al. [51] construisent un graphe de rectangles colorés à partir du document. Les nœuds du graphe sont des rectangles décrits par 4 attributs (I, Q, W, H) dont deux attributs de couleur et deux attributs géométriques et la connexion entre les nœuds du graphe est déterminée par la visibilité entre les rectangles. Deux nœuds sont visibles s'ils partagent une plage horizontale (ou verticale) suffisamment large et s'ils peuvent être connectés sans traverser d'autres rectangles. Enfin, un isomorphisme de sous-graphes est utilisé pour localiser la région d'intérêt dans le document avec un sous-graphe comme requête et un graphe connu comme cible. Enfin, d'autres techniques, telles que PICK [139], TRIE [145] ainsi que [92, 115, 131], fusionnent chaque plongement de graphe de documents avec tous les plongements des tokens dans les segments de texte correspondants. Cela est ensuite introduit dans des modèles d'étiquetage de séquences pour extraire les valeurs des champs recherchés.

2.4.2 Classement par mode d'apprentissage

Les méthodes d'EI peuvent également être classées selon le mode d'apprentissage adopté en trois catégories, comme le montre la Figure 2.2 : l'apprentissage purement supervisé, l'apprentissage par transfert avec une étape de pré-entraînement et l'apprentissage semi-supervisé avec des modèles génératifs.

Apprentissage supervisé

Plusieurs systèmes d'EI appliquent des méthodes d'apprentissage profond purement supervisées [25, 79, 94, 96, 129, 139, 145]. Dans ces modèles, les documents sont représentés sous différents formats tels que des séquences de textes, des graphes ou grilles, comme nous l'avons expliqué précédemment. Chaque modèle peut mettre en relief des aspects différents du document telles que ses caractéristiques textuelles, visuelles ou positionnelles. Le processus d'apprentissage est assuré en une seule étape supervisée. Chaque modèle est entraîné sur un ensemble de documents étiquetés appartenant à un domaine donné. Les principaux défis auxquels sont confrontés

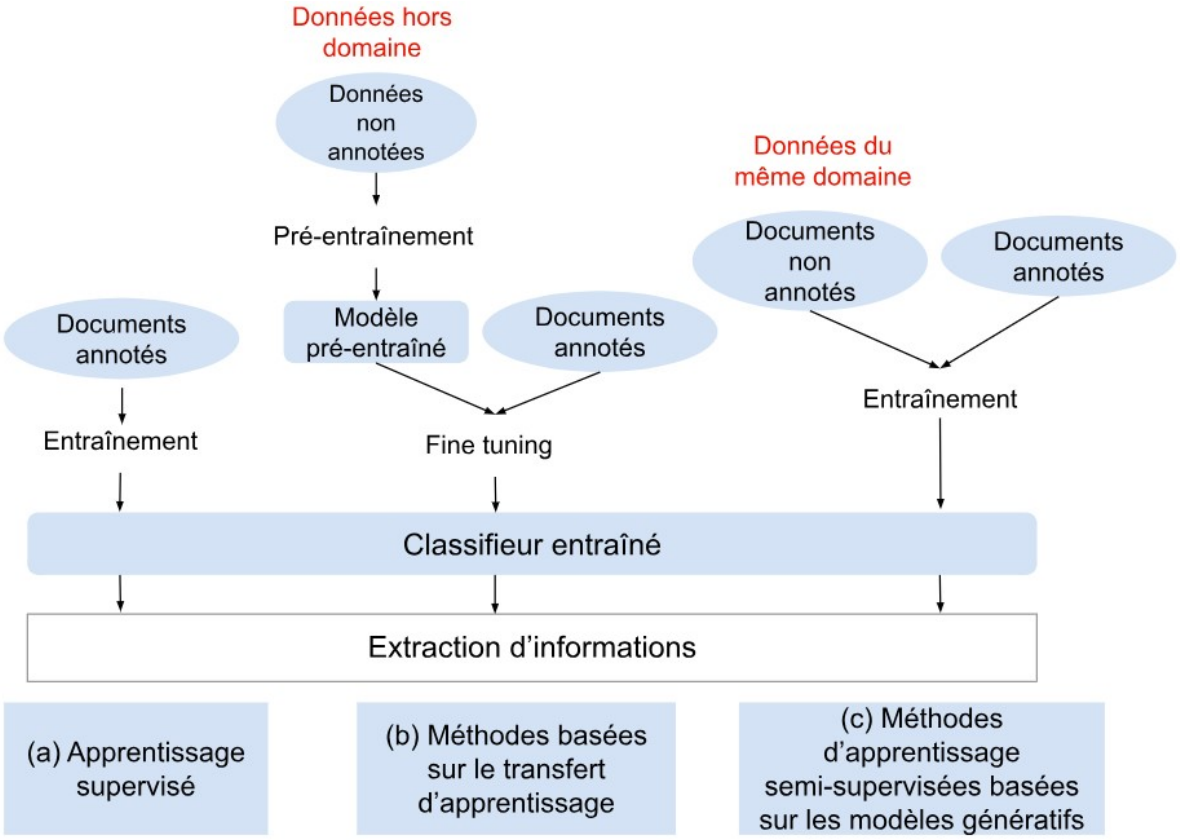


FIGURE 2.2 – Les trois modes d'apprentissage.

ces systèmes sont la grande complexité et le coût de l'étiquetage des documents, ainsi que le manque de jeux de données d'entraînement réels dans le domaine des documents administratifs.

Méthodes basées sur le transfert d'apprentissage

La grande majorité des systèmes d'EI les plus performants tels que [36, 41, 93, 134, 135], appliquent un processus d'entraînement en deux étapes sur des séquences de tokens. La première étape consiste à apprendre les caractéristiques des segments de texte et leur contexte à partir d'un jeu de données non étiquetées à l'aide de diverses tâches de pré-entraînement. La deuxième étape, connue sous le nom de réglage fin ou de « fine tuning », permet au système de se spécialiser dans une tâche particulière en procédant à un apprentissage supervisé à partir de données étiquetées. Le modèle pré-entraîné de la première étape est affiné en ajoutant aux couches pré-entraînées une ou plusieurs couches de sortie qui permettent de classer ou d'étiqueter les tokens. Ces systèmes sont majoritairement basés sur une architecture de Transformer [125] que nous allons détailler à la fin de cette sous-section.

Le modèle de référence dans cette catégorie est BERT [36] qui utilise deux tâches d'apprentissage non supervisé dans l'étape de pré-entraînement, la modélisation du langage masqué et la prédiction de la phrase suivante. La première tâche masque aléatoirement certains tokens d'entrée avec pour objectif de prédire les tokens originaux, tandis que la deuxième tâche est une tâche de classification binaire prenant une paire de phrases comme entrée et les classant comme deux phrases consécutives. Une mise à jour de BERT appelée roBERTa [93] a été ensuite proposée

en éliminant la tâche de prédiction de la phrase suivante. LAMBERT [41] aussi utilise, comme roBERTa, uniquement la tâche de modélisation du langage masqué.

LayoutLM [135], quant à lui, utilise lors du pré-entraînement la tâche de modélisation du langage visuel masqué. Il masque aléatoirement certains des tokens d'entrée tout en conservant leurs plongements de positions 2D, puis il apprend à prédire les tokens masqués en fonction de leurs contextes et positions 2D. De plus, LayoutLM est pré-entraîné sur la tâche de classification des documents en plusieurs classes. LayoutLMv2 [134], pour sa part, effectue le pré-entraînement avec la tâche de modélisation du langage visuel masqué, l'alignement texte-image ainsi que la correspondance texte-image. Pour l'alignement texte-image, certaines lignes contenant des tokens sont sélectionnées au hasard et leurs régions dans l'image sont couvertes. Pendant le pré-entrainement, une couche de classification est ajoutée à la sortie de l'encodeur pour prédire une étiquette pour chaque token selon qu'il est couvert ou non. Concernant la correspondance texte-image, la représentation de sortie du token [CLS] qui est ajoutée à la fin de la séquence, est introduite dans un classifieur afin de prédire si l'image et le texte proviennent de la même page de document.

StrucText [84] incorpore, à son tour, la tâche de prédiction de la longueur du segment où le modèle apprend à reconnaître cette longueur à partir de chaque élément visuel. Le modèle est également pré-entraîné sur la tâche de prédiction des directions des boîtes appariées, ce qui correspond aux relations spatiales entre les paires de segments de texte. Il divise le champ de 360 degrés en huit zones identiques, puis calcule l'angle entre les segments de texte et l'associe à l'une des zones.

L'étape de pré-entraînement dans ces systèmes nécessite une très grande masse de données d'entraînement et les modèles finaux sont considérés comme des modèles complexes (plus de 100 millions de paramètres). Il est donc difficile de les déployer et de les maintenir dans des contextes pratiques.

L'architecture du Transformer : Proposée par Vaswani et al. [125], il repose sur une structure encodeur-décodeur. L'encodeur est chargé de convertir la séquence d'entrée en une série de représentations continues qui sont ensuite transmises au décodeur. Ce dernier utilise à la fois la sortie du décodeur de l'étape temporelle précédente et la sortie de l'encodeur pour générer une séquence de sorties.

L'encodeur est composé de N couches identiques, chacune étant constituée de deux sous-couches. La première sous-couche utilise un mécanisme d'auto-attention multi-têtes. La deuxième sous-couche consiste en un réseau entièrement connecté de type feedforward, composé de deux transformations linéaires avec une activation ReLU entre les deux. En outre, chacune de ces deux sous-couches est entourée d'une connexion résiduelle et suivie d'une couche de normalisation. Des encodages de position sont également introduits dans les plongements d'entrée. Ils sont générés à l'aide de fonctions sinusoïdales et cosinusoïdales de différentes fréquences.

Le décodeur est constitué également de plusieurs couches uniformes, chacune comportant trois sous-couches. La première reçoit la sortie précédente du décodeur ainsi que des informations sur la position et applique le mécanisme d'auto-attention multitêtes masqué pour prédire un mot. Le masque ne prend en compte que les sorties des mots précédents de la séquence. La deuxième couche est similaire à la première sous-couche de l'encodeur. Enfin, la troisième sous-couche présente un réseau entièrement connecté de type feed-forward.

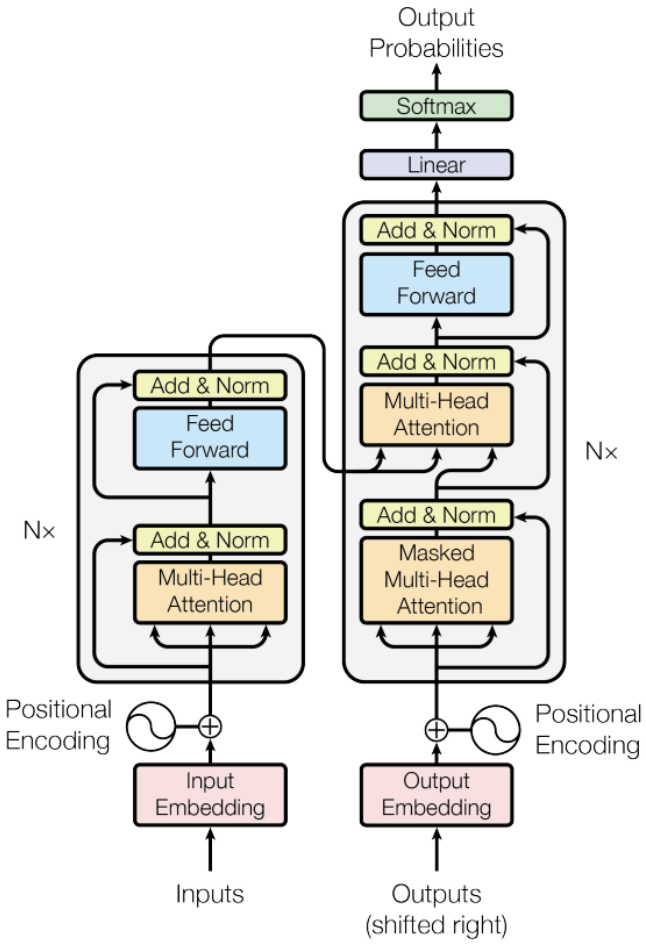


FIGURE 2.3 – L'architecture du Transformer proposée par [125].

Méthodes d'apprentissage semi-supervisées basées sur les modèles génératifs Semi-VAE

Les méthodes génératives profondes ont récemment rendu plus intéressant l'apprentissage semi-supervisé avec les données non étiquetées du même domaine. Elles sont moins complexes que les méthodes basées sur le pré-entraînement et se sont révélées prometteuses dans les applications de classification du texte.

Les approches proposées dans [16, 40, 49, 138] emploient des architectures semi-supervisées basées sur le VAE (Variational Auto-Encoder), un type spécifique des modèles génératifs, pour effectuer des tâches de classification de texte dans des séquences de mots. Ces systèmes se composent généralement de trois éléments principaux, à savoir le classifieur, l'encodeur et le décodeur. Le classifieur fournit une prédiction sur la séquence d'entrée, l'encodeur transforme les données dans un espace latent et le décodeur combine les sorties du classifieur et de l'encodeur pour reconstruire la séquence d'entrée.

Felhi et al. [40] introduisent un VAE semi-supervisé avec un encodeur et un décodeur basés sur une LSTM pour la classification du texte. L'encodeur se compose d'une couche de plongements fastText [12] pré-entraînée et de deux couches BLSTM suivies d'une couche linéaire qui permet la classification des séquence du texte. Afin de simplifier le modèle et d'améliorer la vitesse du processus d'apprentissage, Felhi et al. [40] ont proposé la suppression des composantes associées à

la variable latente ainsi que la perte calculée sur cette variable (la KL-Divergence). Chen et al. [16] ont introduit également un étiqueteur séquentiel variationnel pour l'étiquetage semi-supervisé des séquences. Il s'agit d'un semi-VAE basé sur l'architecture BiGRU [27]. Pour prédire les étiquettes sur la séquence d'entrée à chaque pas de temps, les auteurs introduisent le résultat d'un BiGRU appliqué à la séquence d'entrée dans un réseau neuronal feedforward à une seule couche. De même, Gururangan et al. [49] pré-entraînent un VAE basé sur des réseaux de type feed-forward sur des données non étiquetées. Les représentations apprises sont ensuite concaténées à des vecteurs de mots et introduites à un classifieur. Un autre VAE semi-supervisé assurant la tâche de classification de texte a été proposé par Ye et al. [138]. Tout d'abord, le jeu de données est considéré comme des données non étiquetées pour pré-entraîner le semi-VAE qui consiste en un encodeur et un décodeur basés sur les GCNNs [32], afin d'obtenir des représentations pertinentes. Après la phase de pré-entraînement, les représentations apprises sont introduites dans un MLP pour assurer la tâche de classification.

L'utilisation des VAE semi-supervisés s'est avérée efficace pour d'autres tâches d'analyse de texte telles que l'extraction de relations et l'analyse de sentiments. Par exemple, Zhang et al. [146] présentent un VAE semi-supervisé pour l'extraction des relations biomédicales à partir de textes biomédicaux, basé sur un encodeur BLSTM et un décodeur CNN, avec un classifieur CNN séparé. Le classifieur permet de prédire le type de relation entre deux entités, qui peut correspondre à une relation binaire ou multi-classes (comme conseil, effet, mécanisme, etc.). L'entrée du classifieur et de l'encodeur correspond à une séquence de phrases contenant les deux entités (mots) candidates ainsi que les mots qui les entourent. Chaque mot de la séquence est représenté par la combinaison de son vecteur de plongement fastText [12] avec un encodage de sa position dans la séquence. Cheng et al. [18] présentent un système d'analyse des sentiments à base d'aspects, qui vise à classer la polarité (positive, neutre, négative) du sentiment d'un mot ou d'une phrase à base d'un aspect donné dans un texte. Le modèle proposé est constitué d'un encodeur et d'un décodeur de Transformer [125] et de divers types de classifieurs tels qu'une LSTM et un classifieur basé sur l'attention. Le classifieur prédit la polarité dans une phrase contenant un ensemble d'aspects qui correspondent habituellement à une sous-séquence de la phrase. Par exemple, dans les évaluations d'un restaurant, « service et personnel » peuvent être identifiés comme des aspects à considérer dans le texte.

2.5 Méthodes d'apprentissage dans les réseaux de convolution à graphe

Les caractéristiques spatiales du texte dans les documents administratifs (décrites plus loin dans la section 3.2 du chapitre 2) se prêtent bien à la modélisation par graphe. Nous détaillons dans cette section les différentes approches d'apprentissage à graphe existantes dans la littérature.

Un graphe G représentant un document, peut être défini comme $G = (V, E)$ où V est un l'ensemble des nœuds qui correspondent habituellement à des segments de texte de différentes granularités (caractères, sous-mots, mots, séquences de mots) avec $|V| = m$ où m est le nombre de nœuds dans le graphe. $E \subseteq \{V * V\}$ est l'ensemble des arêtes (i, j) reliant les nœuds $i, j \in V$. G peut aussi être pondéré et défini comme $G = (V, E, W)$ où chaque W_{ij} représente le poids de l'arête $(i, j) \in V$. Les méthodes d'apprentissage basées sur ces structures de graphe répondant à la tâche d'EI peuvent être réparties en deux catégories : l'apprentissage transductif et l'apprentissage inductif.

Dans les approches d'apprentissage transductif, le graphe a une vue sur les noeuds étiquetées (de l'entraînement) et non étiquetées (du test) pendant l'apprentissage. Ces approches ap-

prennent les informations propres aux étiquettes des échantillons étiquetés, puis propagent ces informations aux échantillons non étiquetés par l'intermédiaire du graphe [42, 67]. Une base de données X est constituée d'échantillons étiquetés et non étiquetés tel que $X = \{\{x_i, y_i\}_{i=1}^{l}, \{x_j\}_{j=1}^{u}\}$ où l et u représentent le nombre d'échantillons étiquetés et non étiquetés respectivement, et y_i est l'étiquette de l'échantillon étiqueté. L'algorithme transductif vise à apprendre à prédire les étiquettes des échantillons non étiquetés $\{x_j\}_{j=1}^{u}$ [67]. À partir de ces dernières, un modèle peut éventuellement être dérivé par la suite, si la prédiction de nouveaux éléments de données non vus est nécessaire [42]. La plupart des approches de classification de texte utilisant les graphes adoptent l'apprentissage transductif. Notamment, toutes les méthodes proposées dans [89, 132, 137, 144, 152] sont transductives et se basent sur le modèle GCN [76] pour effectuer la tâche de classification. TextGCN [137] a construit un graphe pour l'ensemble du corpus de textes avec les documents et les mots comme nœuds. Il capture donc les informations globales d'un corpus entier et procède à la classification des mots et des documents. SGC [132] et S^2GC [152] ont construit un graphe comme TextGCN, mais ont proposé des approches différentes de calcul de la convolution dans le graphe (propagation de l'information). TG-Transformer [144] et BERTGCN [89] quant à eux, ont appliqué un transformer avec des plongements GloVe pré-entraînés et BERT, respectivement sur le graphe proposé dans TextGCN [137].

Les données non étiquetées dans ce type d'apprentissage, peuvent contribuer à améliorer les performances de la classification dans le graphe. Néanmoins, cet algorithme souffre de la difficulté de construire un modèle générique capable de traiter de nouveaux échantillons qui ne sont pas inclus dans l'échantillon de test non étiqueté de départ. L'utilisation de cette catégorie de modèles d'apprentissage, peut ainsi nécessiter un espace de calcul relativement important lorsque la taille du corpus est grande [130].

Les méthodes basées sur les graphes inductifs peuvent entraîner un classifieur à l'aide d'échantillons étiquetés uniquement ou avec des échantillons non étiquetés. Ce classifieur est capable d'assurer la tâche de la classification sur de nouveaux échantillons non étiquetés [50, 85]. Étant donné la base de données composée d'échantillons étiquetés et non étiquetés, X, l'algorithme inductif vise à apprendre à prédire l'étiquette pour n'importe quelle entrée $x \in X$. L'objectif de l'apprentissage inductif est de pouvoir prédire les étiquettes des nouveaux échantillons via la structure du graphe lorsque les nouveaux échantillons (non vus) entrent dans la structure du graphe [26]. Plusieurs modèles de classification de texte basés sur des graphes construisent ainsi un graphe pour chaque document en utilisant les mots comme nœuds [58, 110, 147]. Peng et al. [110] représentent les noeuds des mots par des plongements Word2Vec pré-entraînés et construisent les arêtes en utilisant des informations sur la co-occurrence des mots, puis ils utilisent des CNN pour la classification des mots. Huang et al. [58] représentent les mots par leurs plongements GloVe pré-entraînés et les connectent à un nombre limité de mots consécutifs dans la même phrase. Ils utilisent ensuite une méthode appelée mécanisme de passage de messages (MPM) [46] pour calculer la convolution et classer le texte. Zhang et al. [147] ont proposé TextING où les connexions entre les mots s'appuient sur leurs co-occurrences dans une fenêtre de taille fixe (3). Sur chaque graphe construit, ils emploient les réseaux GGNN (Gated Graph Neural Networks) [83] pour apprendre les plongements des mots tout en prenant en considération leurs voisins. Deux perceptrons multicouches (MLP) sont utilisés afin de regrouper les représentations apprises par les GGNN. Enfin, l'étiquette est prédite à la sortie à l'aide d'une couche Softmax.

3

Système semi-supervisé pour l'extraction d'informations dans les DSSs

Sommaire

3.1 Introduction

Pour permettre l'extraction d'informations à partir d'un DSS, nous proposons le pipeline décrit dans la Figure 3.1. Nous avons opté pour une approche opérant au niveau du mot. En effet, tous les composants élémentaires de ce dernier (caractères, n-grammes de caractères, sous-mots) partagent régulièrement les mêmes caractéristiques positionnelles et appartiennent à la même classe d'information. Afin de mieux exploiter les caractéristiques des DSSs illustrées plus loin dans la section 3.2 de ce chapitre, nous avons adopté une modélisation des DSSs sous forme de graphe de mots. Cette représentation est considérée comme la plus adaptée à ce type de document, comparée aux séquences de tokens et aux grilles 2D, car elle permet de mettre en valeur les relations spatiales entre les mots qui sont essentielles dans un DSS.

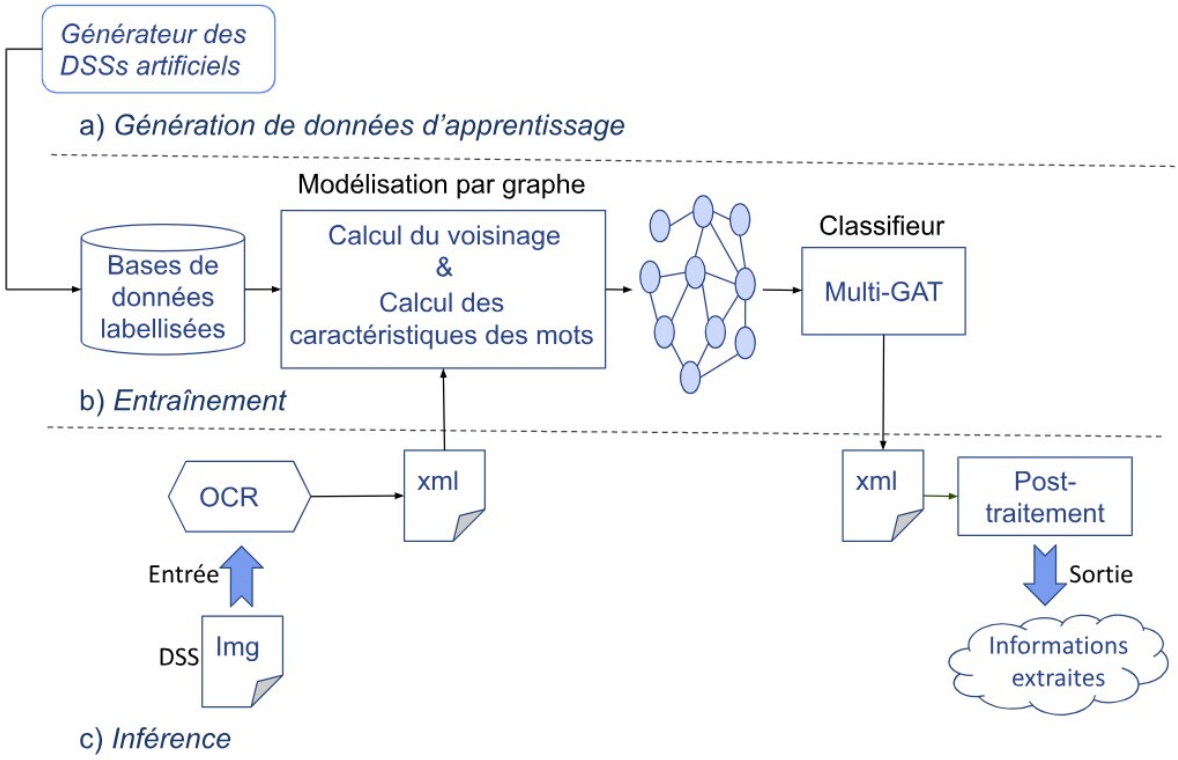


FIGURE 3.1 – Le pipeline d'EI proposé, décrit en trois étapes.

Comme le montre la Figure 3.1, le système développé est composé de trois étapes : génération de documents, entraînement et inférence. La première étape est réservée à la génération artificielle de documents afin de disposer d'assez d'échantillons annotés pour les besoins de l'apprentissage et du test du système. Ces documents artificiels peuvent se suffire à eux-mêmes ou venir compléter des bases existantes (publiques ou privées). À l'étape d'entraînement, tout document issu de ces bases, est modélisé par un graphe où les noeuds représentent les mots et les arcs, les relations de voisinage entre eux. Les mots sont décrits dans les nœuds, chacun par un

vecteur de caractéristiques. Le graphe obtenu est ensuite introduit dans un classifieur de nœuds du graphe, appelé Multi-GAT. À l'étape d'inférence, le document est passé dans un module de reconnaissance optique de caractères (OCR) en vue d'extraire le texte et les boîtes englobantes des mots. Le résultat est décrit en « xml », format également utilisé par le générateur. Les mêmes étapes de modélisation par un graphe et de classification des nœuds sont utilisées, suivies d'un post-traitement en vue de regrouper les mots appartenant à la même classe et retourner les entités finales.

Dans les différentes sections de ce chapitre, nous décrirons les approches utilisées pour modéliser le DSS et mettre en œuvre le classifieur.

Tout d'abord, nous présentons l'approche de génération de DSSs et d'évaluation de leur diversité. Ensuite, nous décrivons une version supervisée du Multi-GAT transductif proposé avec un graphe regroupant tous les mots du corpus, que nous noterons « $Multi-GAT_{Dataset}$ ». Puis, nous présentons une deuxième version inductive améliorée « $Multi-GAT_{DSS}$ ». Cette version modélise les documents par des graphes séparés et enrichit les caractéristiques des mots dans le graphe.

Un autre modèle semi-supervisé est enfin présenté. Ce modèle correspond à un Multi-GAT inductif semi-supervisé noté « Mutli-GAT+VGAE ». Ce modèle est capable d'apprendre à partir des DSSs annotés et non annotés simultanément à l'aide du modèle génératif « VGAE : Variational Graph Auto Encoder ».

3.2 Préparation et génération des données d'entraînement

3.2.1 Motivations

Les documents administratifs contenant des données personnelles, sont des documents confidentiels, ce qui soulève une contrainte très importante pour la mise en œuvre de modèles neuronaux traitant ce type de documents, à savoir le manque de données d'entraînement. Pour entraîner les modèles et évaluer leurs performances, nous avons besoin de jeux de données de documents annotés. En raison du manque de jeux de données de documents administratifs, nous avons commencé par proposer et développer un générateur de documents semi-structurés.

3.2.2 État de l'art sur la génération de données d'apprentissage

La génération de données synthétiques, notamment les documents, pour entraîner ou améliorer les performances des modèles neuronaux, a été considérablement étudiée dans plusieurs travaux ces dernières années [113].

Capobianco et al. [15] ont proposé l'outil DocEmul qui est capable de générer des documents historiques synthétiques qui imitent une collection réelle de manuscrits structurés. À partir de quelques échantillons de pages, ils extraient les fonds des pages à l'aide d'algorithmes de binarisation. Ces fonds sont ensuite utilisés comme supports d'écriture, avec des structures variables. Celles-ci sont créées à partir de fichiers XML contenant les configurations essentielles telles que la structure de l'en-tête, les polices de caractères, les dictionnaires et les éléments visuels. Raman et al. [117], quant à eux, génèrent des articles scientifiques synthétiques. Ils proposent une méthode de génération traitant chaque composant physique du document comme une variable aléatoire et modélisant leurs dépendances à l'aide d'un graphe de réseau bayésien. Ce réseau permet de définir le processus de génération des documents. Les styles, les polices et les éléments de mise en page sont tous traités comme des variables aléatoires et représentés comme des nœuds dans le graphe. Ces variables sont considérés comme des informations d'entrée dans le modèle DDR proposé dans

[90]. Ce dernier fournit des pages d'articles scientifiques fictives en modélisant aléatoirement le contenu textuel et non textuel importé de sources d'information en ligne. Sánchez et al. [121] génèrent une base de données des factures d'électricité espagnoles. Sur la base d'un ensemble de factures provenant de plusieurs fournisseurs, ils créent un grand nombre de documents contenant des données fictives. Avant de générer la base de données, ils collectent un certain nombre de dictionnaires contenant des noms et prénoms espagnols, des institutions financières, des villages et des rues en Espagne, etc. Ensuite, ils créent un ensemble de documents modèles (templates) basés sur des factures réelles. Dans un premier temps, des données aléatoires sont générées à partir des dictionnaires et de plusieurs statistiques liées aux principaux montants des factures, ce qui donne lieu à un fichier xml. Ce fichier est alors utilisé pour remplir le modèle et générer la facture finale.

Plusieurs autres modèles neuronaux génératifs ont été proposés pour générer des mises en page de documents à partir d'un jeu de données de documents. Biswas et al. [10] proposent d'apprendre des mises en page dans les documents à partir d'un ensemble d'images, en se basant sur le réseau GAN (Generative Adversarial Network). Patil et al. [108] proposent également une architecture basée sur le modèle génératif VAE (Variational Auto-Encoder) et RNN (Recurrent Neural Network). Tout d'abord, les mises en page d'entraînement sont transformées en une représentation latente, correspondant à une distribution gaussienne. De nouvelles mises en page peuvent dès lors être générées en échantillonnant de nouvelles valeurs de cette distribution. Une méthode basée sur les GAN est également proposée par Zheng et al. [151] pour générer des mises en page de documents à partir d'images, de mots-clés et de catégories de documents. Le problème de cette approche est qu'elle nécessite une grande quantité de données d'apprentissage. Nous trouvons également LayoutTransformer [48] qui tire parti du mécanisme d'attention pour apprendre les relations contextuelles entre les éléments de la mise en page et générer de nouvelles mises en page dans un domaine précis. LayoutTransformer prend des éléments de mise en page en entrée et prédit les éléments suivants en sortie. Pendant l'apprentissage, il utilise la vérité terrain des tokens de mise en page comme entrée du bloc de décodage. La première couche de ce bloc est une couche d'attention masquée qui permet au modèle de ne voir que les éléments précédents afin de prédire l'élément actuel. Les auteurs de [113] exploitent aussi cette architecture de LayoutTransformer pour générer des articles scientifiques. Ils entraînent le modèle génératif (basé sur LayoutTransformer) et produisent de grandes collections de pages avec une mise en page variable où les régions sont remplies de contenu généré synthétiquement (avec du texte, des images et des tableaux). Ils utilisent NLTK (Natural Language Tool Kit) [9] pour générer le texte et ils collectent les images, les tableaux et les formules sur Internet.

Enfin, nous citons le travail de Blanchard et al. [11] qui a donné lieu à la première version du générateur des documents administratifs implémenté par l'équipe READ et la société Fair&Smart. Ils génèrent deux types de factures : des modèles fixes (voir Figure 3.2) et des modèles aléatoires. Les modèles fixes sont une combinaison statique d'éléments physiques qui permettent à l'utilisateur de générer plusieurs factures avec des données variées. Cette méthode reproduit des mises en pages de factures réelles existantes. Elle produit une vérité terrain très proche de la réalité, mais nécessite une implémentation spécifique pour chaque catégorie de facture. Les modèles aléatoires, quant à eux, proviennent d'une structure générique de la facture comportant des parties pouvant avoir un comportement aléatoire. Des zones globales ont été créées, comprenant l'en-tête, le tableau des produits, le pied de page des produits et le pied de page global. Ces zones comprennent les informations sur la société, la commande, les produits, l'adresse de facturation et l'adresse de livraison. Tous ces éléments sont placés de manière aléatoire dans la page.

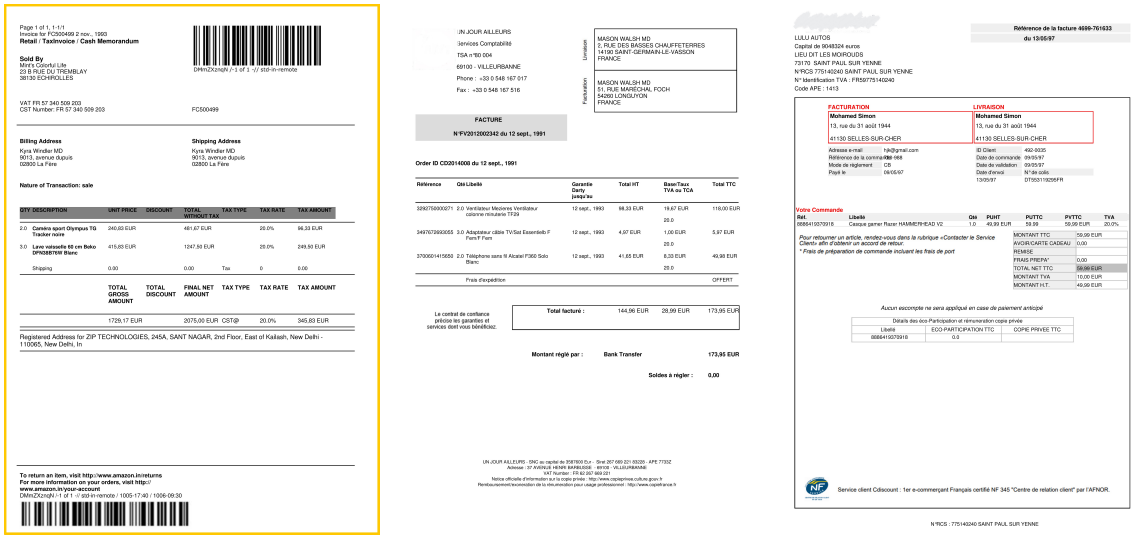


FIGURE 3.2 – Exemples de modèles fixes de factures générés par [11].

3.2.3 Modélisation des documents semi-structurés (DSSs)

Définition des DSSs

D'une manière générale, un document peut appartenir à l'une des trois catégories suivantes : document structuré, document semi-structuré et document non structuré que nous pouvons décrire comme suit :

— Un document structuré conserve la même mise en page : les champs ou les informations sont toujours positionnés aux mêmes endroits dans la page et leur nombre et type sont toujours inchangés. Des expressions régulières ou des modèles (templates) prédéfinis peuvent être utilisés pour extraire les informations de ce type de document.

— Un document non structuré, quant à lui, contient des informations intégrées dans du texte libre ou dans une séquence de paragraphes qui, pour être extraites, nécessitent l'utilisation de techniques de TAL.

— Un document semi-structuré, enfin, est un document dans lequel l'emplacement des données et les champs contenant les informations utiles à extraire, varient d'un document à l'autre. Par exemple, l'adresse de livraison dans une facture peut se trouver en haut à gauche, en haut au milieu ou en bas à droite de la page. Dans les tableaux de produits d'une facture, le nombre de champs, de lignes ou d'articles par transaction, peuvent également varier d'une facture à l'autre et d'un fournisseur à l'autre. Certaines informations, voire des colonnes entières dans les tableaux, peuvent être facultatives, présentes dans certains documents mais absentes dans d'autres du même type. Les éléments de contenu peuvent prendre trois formes : accompagnés de mots-clés, bien formatés sans mots-clés, comme les dates, ou regroupés sur plusieurs lignes, contenant quelques mots-clés indicatifs comme « rue » ou « code postal » dans le cas d'une adresse.

Les caractéristiques de la mise en page des DSSs peuvent être résumées par les deux points suivants :

— Il n'y a pas de contraintes fixes sur la mise en page ;

— Les éléments ne se positionnent pas aux mêmes endroits dans les DSSs appartenant à la même catégorie.

Quant aux caractéristiques des informations dans les DSSs, elles peuvent être synthétisées comme suit :

— Elles peuvent être accompagnées de mots-clés indicatifs séparés par deux points (:) ou un espace ;

— Elles peuvent avoir un format très spécifique sans être accompagnées par des mots-clés comme les dates ;

— Elles sont obligatoires ou optionnelles.

Description de la structure des DSSs

Bien qu'il n'existe pas de définition précise des DSSs, nous en avons analysé un grand nombre pour proposer un modèle générique qui nous permet de produire des documents aussi proches que possible de la réalité et en nombre suffisant.

Comme illustré dans la Figure 3.3, nous modélisons une page DSS comme une séquence verticale de blocs (B1,B2,...). Chacun de ces blocs est une séquence horizontale de sous-blocs que nous appelons groupes (G1,G2,...). Chaque groupe est, à son tour, une séquence verticale de composants élémentaires (EC : Elementary Components). Un EC peut contenir une paire (mots-clés (KW : keywords), information), une information sans mots-clés ou un tableau.

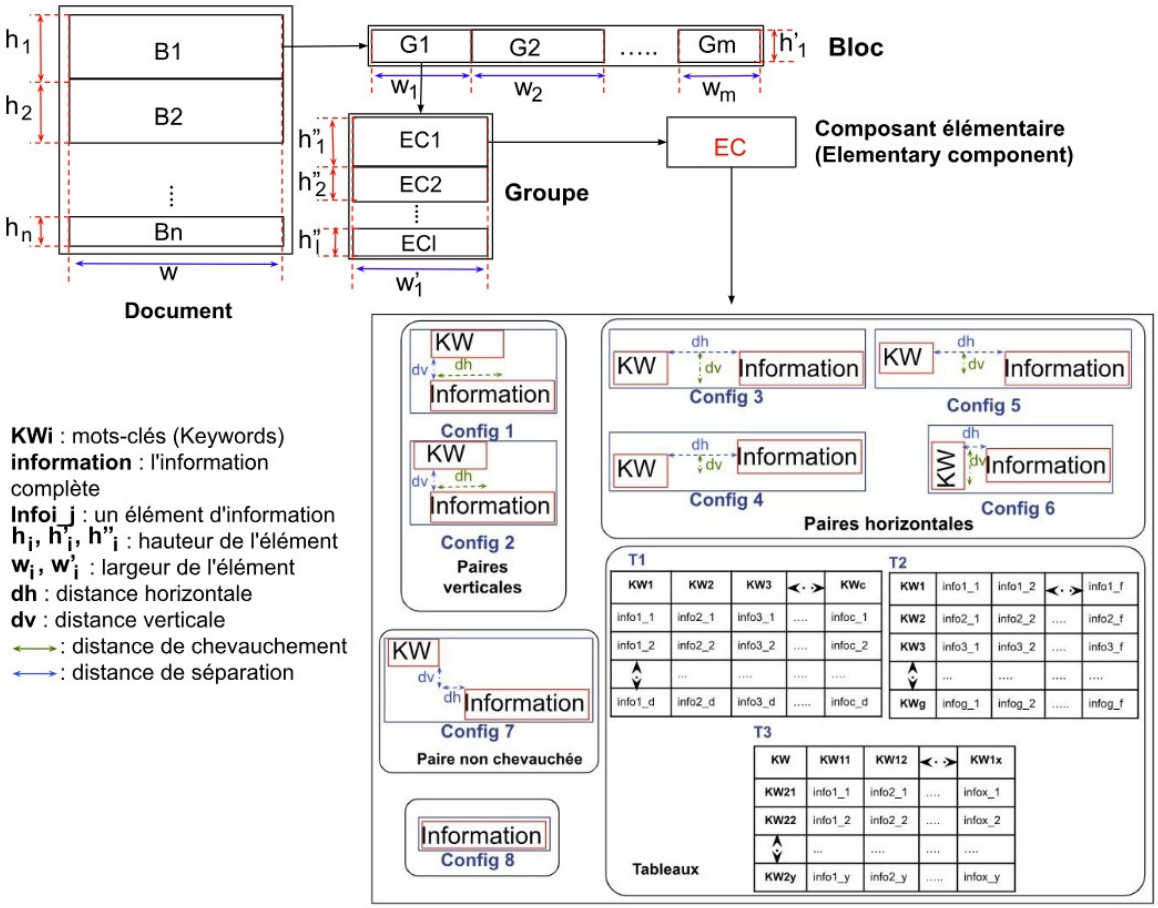


FIGURE 3.3 – La structure proposée pour un DSS.

Comme le montre la Figure 3.3, sept configurations (Config 1 à Config 7) sont possibles pour les paires, correspondant à différentes positions des mots-clés par rapport à l'information (horizontale, verticale ou autre). Un EC peut également ne contenir que des informations sans mots-clés (Config 8) ou un tableau (T1, T2 ou T3). La séquence des blocs formant la page du DSS est centrée dans la page. Le contenu des blocs, des groupes et des EC peut être justifié verticalement (en haut, au centre ou en bas) et horizontalement (à droite, à gauche et au centre).

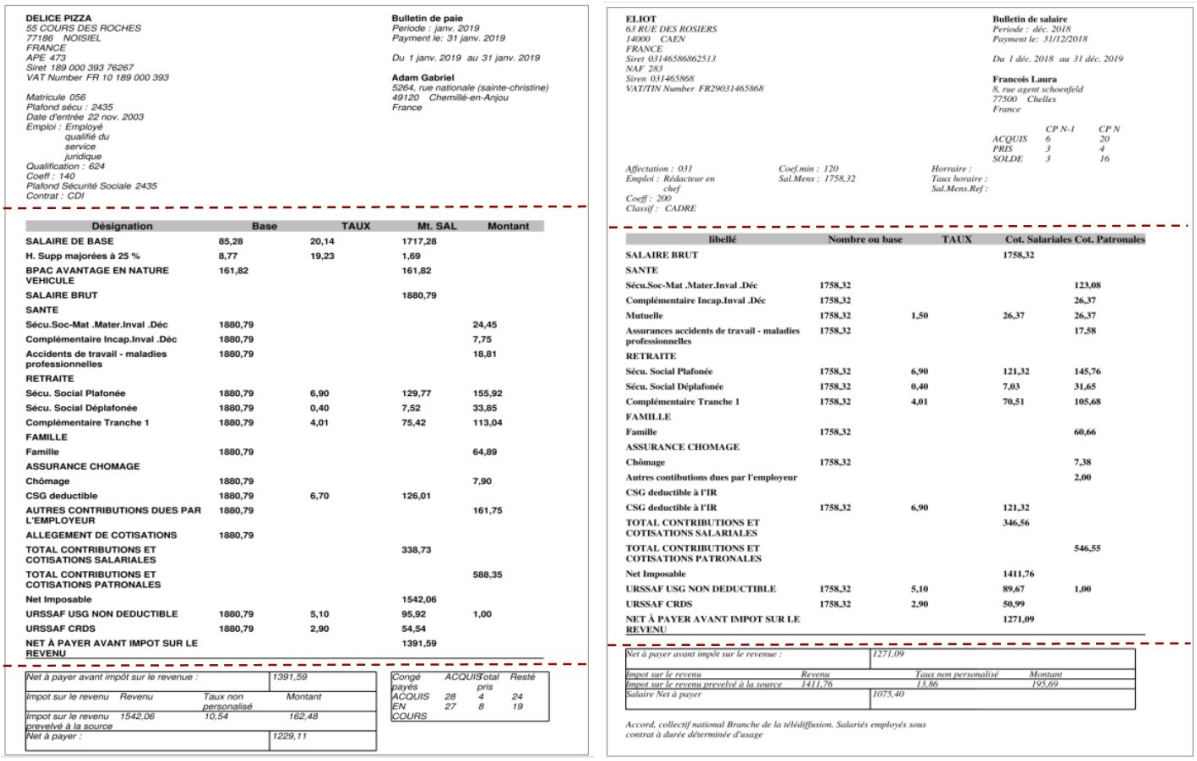
Nous proposons une approche de génération automatique de DSSs en se basant sur ces différentes configurations et sur des variables aléatoires pour gérer le contenu et l'emplacement des différents blocs dans les documents.

3.2.4 Les trois cas d'utilisation étudiés

Nous nous sommes intéressés tout particulièrement à l'étude de trois types de DSS à savoir les fiches de paie françaises, les tickets de caisse et les factures françaises et anglaises. Dans cette section, nous montrons la modélisation proposée pour ces trois types de DSS et nous détaillons également les informations obligatoires et facultatives pour chaque type de DSS.

Fiches de paie

La fiche de paie française est divisée en trois parties principales : les informations relatives à l'employeur et à l'employé, les composantes du salaire et les cotisations salariales et en dernier, la synthèse des informations importantes, comme le montre la Figure 3.4.



DELICE PIZZA
55 COURS DES ROCHES
77186 NOISIEL
FRANCE
APE 473
Siret 189 000 393 76267
VAT Number FR 10 189 000 393

Matricule 056
Plafond sécu : 2435
Date d'entrée 22 nov. 2003
Emploi : Employé qualifié du service juridique
Qualification : 624
Coeff : 140
Plafond Sécurité Sociale 2435
Contrat : CDI

Bulletin de paie
Periode : janv. 2019
Payment le: 31 janv. 2019

Du 1 janv. 2019 au 31 janv. 2019

Adam Gabriel
5264, rue nationale (sainte-christine)
49120 Chemillé-en-Anjou
France

Désignation	Base	TAUX	Mt. SAL	Montant
SALAIRE DE BASE	85,28	20,14	1717,28	
H. Supp majorées à 25 %	8,77	19,23	1,69	
BPAC AVANTAGE EN NATURE VEHICULE	161,82		161,82	
SALAIRE BRUT			1880,79	
SANTE				
Sécu.Soc-Mat .Mater.Inval .Déc	1880,79			24,45
Complémentaire Incap.Inval .Déc	1880,79			7,75
Accidents de travail - maladies professionnelles	1880,79			18,81
RETRAITE				
Sécu. Social Plafonée	1880,79	6,90	129,77	155,92
Sécu. Social Déplafonée	1880,79	0,40	7,52	33,85
Complémentaire Tranche 1	1880,79	4,01	75,42	113,04
FAMILLE				
Famille	1880,79			64,89
ASSURANCE CHOMAGE				
Chômage	1880,79			7,90
CSG deductible	1880,79	6,70	126,01	
AUTRES CONTRIBUTIONS DUES PAR L'EMPLOYEUR	1880,79			161,75
ALLEGEMENT DE COTISATIONS	1880,79			
TOTAL CONTRIBUTIONS ET COTISATIONS SALARIALES			338,73	
TOTAL CONTRIBUTIONS ET COTISATIONS PATRONALES				588,35
Net Imposable			1542,06	
URSSAF USG NON DEDUCTIBLE	1880,79	5,10	95,92	1,00
URSSAF CRDS	1880,79	2,90	54,54	
NET À PAYER AVANT IMPOT SUR LE REVENU			1391,59	

Net à payer avant impôt sur le revenue :		1391,59
Impot sur le revenu Revenu	Taux non personalisé	Montant
Impot sur le revenu prevelvé à la source 1542,06	10,54	162,48
Net à payer :		1229,11

Congé payés	ACQUIS	Total pris	Resté
ACQUIS	28	4	24
EN COURS	27	8	19

ELIOT
63 RUE DES ROSIERS
14000 CAEN
FRANCE
Siret 03146586862513
NAF 283
Siren 031465868
VAT/TIN Number FR29031465868

Bulletin de salaire
Periode : déc. 2018
Payment le: 31/12/2018

Du 1 déc. 2018 au 31 déc. 2019

Francois Laura
8, rue agent schoenfeld
77500 Chelles
France

	CP N-1	CP N
ACQUIS	6	20
PRIS	3	4
SOLDE	3	16

Affectation : 031
Emploi : Rédacteur en chef
Coeff : 200
Classif : CADRE
Coef.min : 120
Sal.Mens : 1758,32
Horraire :
Taux horaire :
Sal.Mens.Ref :

libellé	Nombre ou base	TAUX	Cot. Salariales	Cot. Patronales
SALAIRE BRUT			1758,32	
SANTE				
Sécu.Soc-Mat .Mater.Inval .Déc	1758,32			123,08
Complémentaire Incap.Inval .Déc	1758,32			26,37
Mutuelle	1758,32	1,50	26,37	26,37
Assurances accidents de travail - maladies professionnelles	1758,32			17,58
RETRAITE				
Sécu. Social Plafonée	1758,32	6,90	121,32	145,76
Sécu. Social Déplafonée	1758,32	0,40	7,03	31,65
Complémentaire Tranche 1	1758,32	4,01	70,51	105,68
FAMILLE				
Famille	1758,32			60,66
ASSURANCE CHOMAGE				
Chômage	1758,32			7,38
Autres contibutions dues par l'employeur				2,00
CSG deductible à l'IR				
CSG deductible à l'IR	1758,32	6,90	121,32	
TOTAL CONTRIBUTIONS ET COTISATIONS SALARIALES			346,56	
TOTAL CONTRIBUTIONS ET COTISATIONS PATRONALES				546,55
Net Imposable			1411,76	
URSSAF USG NON DEDUCTIBLE	1758,32	5,10	89,67	1,00
URSSAF CRDS	1758,32	2,90	50,99	
NET À PAYER AVANT IMPOT SUR LE REVENU			1271,09	

Net à payer avant impôt sur le revenue :		1271,09
Impot sur le revenu Revenu	Taux non personalisé	Montant
Impot sur le revenu prevelvé à la source 1411,76	13,86	195,69
Salaire Net à payer	1075,40	

Accord, collectif national Branche de la télédiffusion. Salariés employés sous contrat à durée déterminée d'usage

FIGURE 3.4 – Exemples de fiches de paie générées. Elles sont divisées en trois parties principales visualisées par une ligne tiretée rouge, mais le contenu et les détails de la mise en page de chaque partie sont très variés.

La première partie peut être divisée en un ou deux blocs. Elle regroupe les informations relatives à l'entreprise, les dates de paiement et les informations relatives à l'employé. Elle peut aussi contenir les informations relatives aux congés (sous forme de tableau) ainsi que le nom de la convention collective nationale. La deuxième partie est représentée par un tableau qui contient les informations sur le salaire et les différentes cotisations salariales et patronales. La troisième partie résume les informations importantes du salaire : le brut, le net, les impôts, etc. et peut également contenir les informations sur les congés et la convention collective nationale si elles n'apparaissent pas dans la première partie. Nous détaillons dans la Table 3.1 toutes les informations obligatoires et facultatives que l'on peut trouver dans une fiche de paie.

TABLE 3.1 – Informations obligatoires et facultatives dans les fiches de paie.

	Employeur	Employé	Composant salarial	Autres
Obligatoire	Nom Adresse Code APE Code NAF Numéro Siret Contributions	Nom Adresse Emploi Niveau/Coefficient Cotisations	Salaire brut Salaire net	Période Nombre d'heures Date de paiement Convention collective Informations sur les congés
Optionnel	Logo Téléphone Fax RCS VAT	Affectation Type de contrat Ancienneté Qualification Coefficient/index Plafond de la sécurité sociale Numéro de sécurité sociale Salaire mensuel Salaire de référence	Net avant impôts	Mode de paiement Date

Tickets de caisse

Un ticket de caisse est un DSS qui prouve un paiement. Ce document a une mise en page variable en raison de la variabilité des entreprises et des logiciels qui produisent leurs tickets de caisse (voir Figure 3.5).

La plupart des tickets de caisse peuvent être divisés en trois parties : la première partie contient les informations relatives à l'entreprise, les références du ticket, la date et l'heure, le nom du client, etc. La deuxième partie est réservée au tableau des produits vendus et la dernière partie résume les valeurs des totaux, les taxes, le montant du paiement, le mode de paiement et les notes de bas de page. La première et la troisième partie peuvent être composées de plusieurs blocs tandis que la deuxième contient un bloc représenté par un tableau de produits. Les informations détaillées, obligatoires et facultatives qui figurent généralement dans les tickets de caisse, sont résumées dans la Table 3.2. D'autres informations peuvent être trouvées dans certains tickets de caisse : les détails des produits vendus ou du service. Par exemple, les restaurants peuvent ajouter le numéro de la table, le nom du serveur, etc.

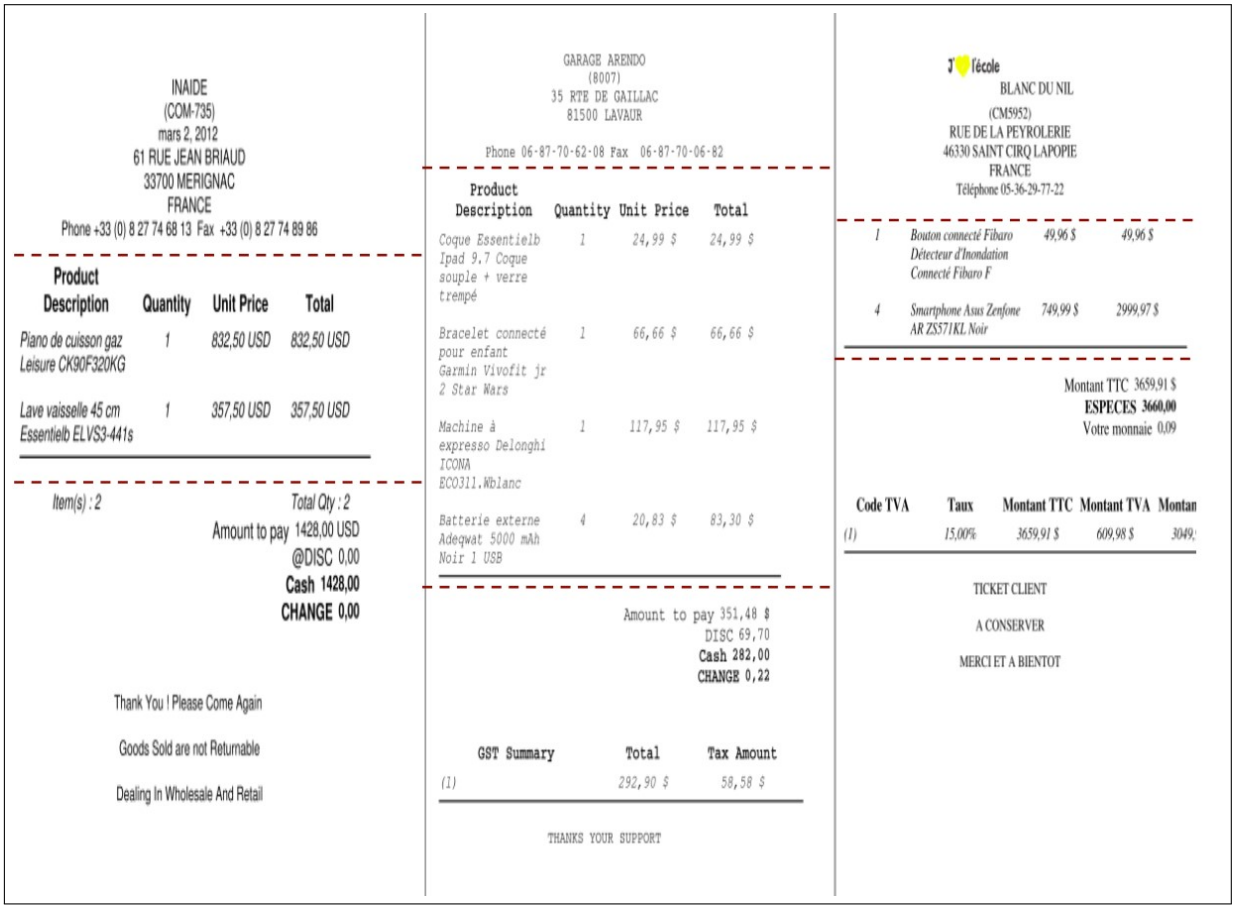


FIGURE 3.5 – Exemples de tickets de caisses générés. Ils sont divisés en trois parties principales, visualisées par des lignes tiretées rouge, dont le contenu et les détails de la mise en page sont variés.

TABLE 3.2 – Informations obligatoires et facultatives dans les tickets de caisse.

	Société	Produits vendus	Autres
Obligatoire	Nom Adresse	Liste de noms Prix Montant total payé Taxe sur les ventes Réductions	Date et heure de la transaction Montant du paiement
Optionnel	Logo Informations de contact Numéro de SIRET Numéro de SIREN Numéro de TVA		Mode de paiement Numéros de référence des tickets Politique de retour Conditions générales de vente

Factures

Les factures sont des documents très similaires aux tickets de caisse. La principale différence entre les deux types de documents est la suivante : une facture est une demande de paiement

émise par l'entreprise avant le paiement, tandis qu'un ticket est une preuve de paiement. Les factures contiennent des informations supplémentaires relatives au client, telles que les adresses de livraison et de facturation. En revanche, elles ne contiennent généralement pas d'informations relatives au paiement (espèces, monnaie, etc.). Elles ont le plus souvent un format A4, contrairement aux tickets de caisse qui peuvent avoir des dimensions différentes.

Une facture (française ou anglaise) est divisée en 3 parties : la première contient les informations de l'entreprise ainsi que les informations de livraison et de facturation ; la deuxième partie est réservée au tableau des produits ; la dernière partie contient les totaux et les notes de bas de page. Nous pouvons voir cela dans les exemples de la Figure 3.6.

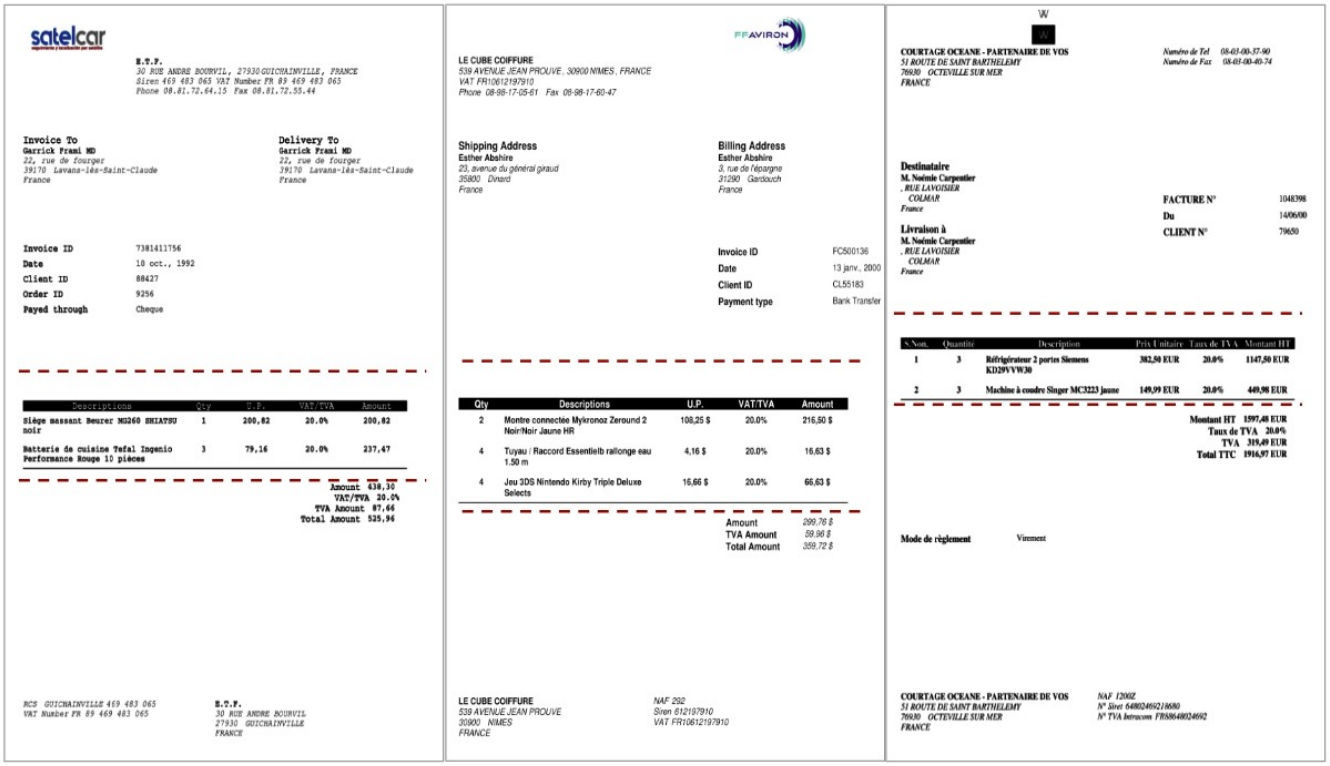


FIGURE 3.6 – Exemples de factures générées. Elles sont divisées en trois parties principales, visualisées par des lignes tiretées rouge, dont le contenu et les détails de la mise en page sont variés.

3.2.5 Variation du contenu et de la mise en page dans un DSS

Nous détaillions dans cette section la méthode proposée pour la génération des DSSs.

La création d'un DSS comporte deux étapes distinctes. La première consiste à générer le contenu des trois types de DSS et la seconde à organiser ce contenu sur le document, en gérant sa mise en page. Cette section décrit ces deux étapes en détail.

La variation du contenu

Comme le montre la Figure 3.7, un modèle générique (« Model ») est créé pour générer le contenu des trois types de DSS. Il rassemble les propriétés communes des trois types de DSS, à savoir : les informations relatives à l'entreprise et au client. Il contient également la langue du

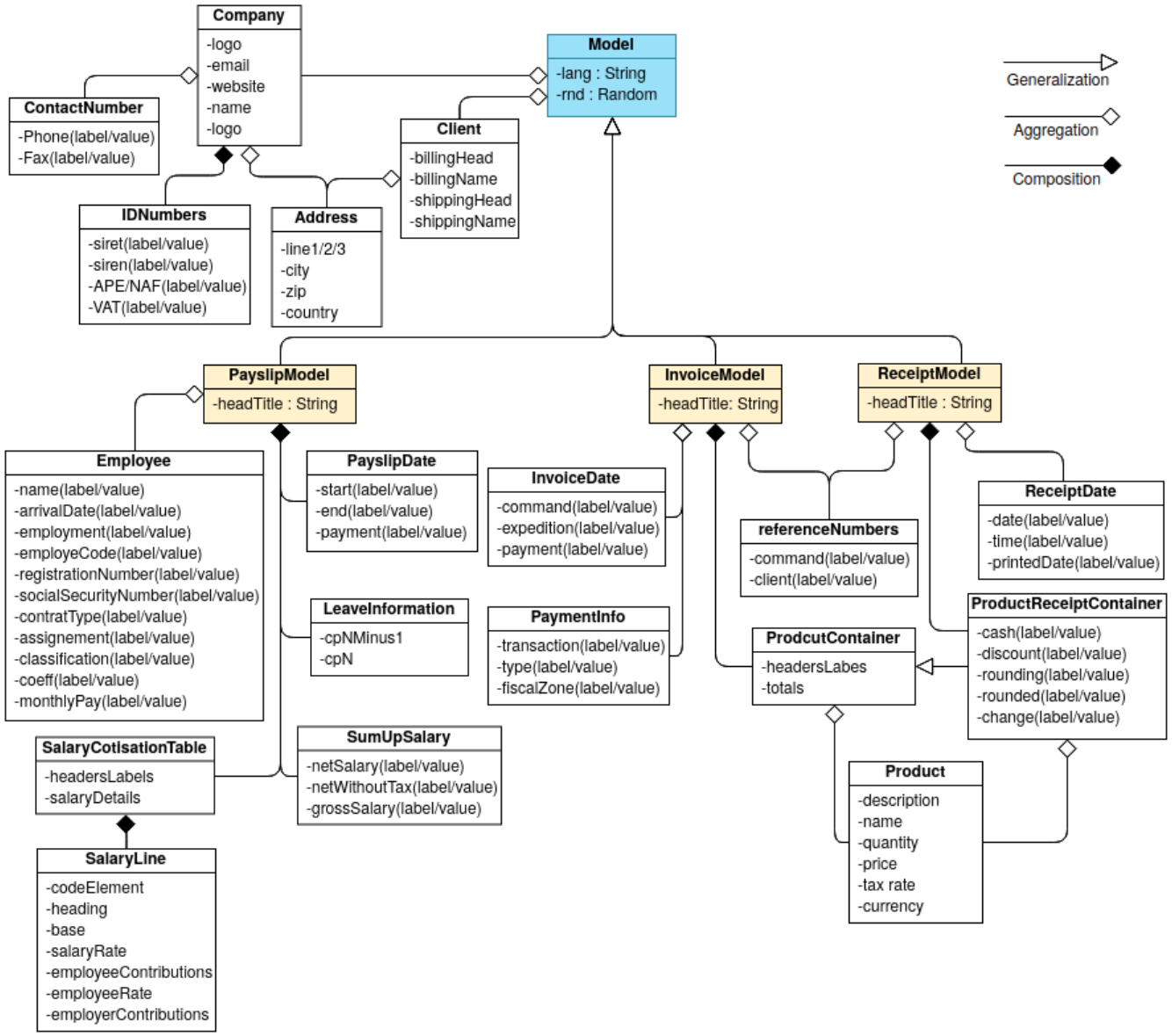


FIGURE 3.7 – Représentation UML du diagramme de classes de la génération de contenu DSS : l'association de composition est utilisée lorsqu'une classe fait partie d'une autre classe et ne peut exister sans elle (IDNumbers et Company), tandis que la classe enfant dans l'agrégation peut exister indépendamment de sa classe mère (Address et Client).

document (lang), son entête (headTitle) et une variable aléatoire (rnd) utile pour la gestion du contenu.

Nous créons ensuite, pour chaque type de DSS, son propre modèle qui instancie ses propriétés spécifiques (« PayslipModel », « InvoiceModel » et « ReceiptModel »). Les différents types de DSS peuvent partager des propriétés. Par exemple, les factures et les tickets de caisse partagent les informations sur les produits et les références. Comme le montre la Figure 3.7, la plupart des attributs des classes sont représentés par leur étiquette et leur valeur. Cela correspond au contenu de l'EC dans le DSS : la paire (mots-clés (KW), information), où les étiquettes sont les mots-clés et les valeurs sont les informations.

Pour générer un nouveau type de DSS, nous pouvons étendre le schéma actuel en ajoutant de nouvelles classes si nécessaire (tableau des paiements dans un document de relevé bancaire par exemple) et exploiter les classes existantes que l'on peut trouver dans la plupart des documents administratifs (société, client, date, etc.).

Stratégie de variation du contenu Des variables aléatoires sont chargées de gérer la génération du contenu de chaque DSS, conformément au schéma de la Figure 3.7. L'attribut « rnd » de la classe « Model » est utilisé pour faciliter cette tâche. Pour générer les mots-clés, une variable aléatoire est utilisée pour sélectionner des mots-clés au hasard dans des listes. Les variations de mots-clés étant assez limitées, des listes de mots-clés sont créées et remplies dans les corps des différentes classes.

Pour générer la partie information, différents types de variables aléatoires viennent gérer cette partie, en fonction du type et de la nature de l'information :

— Si l'information est une valeur numérique ou une date : la variable aléatoire prend une valeur numérique aléatoire en fonction de seuils définis ou de formats précis prédéfinis.

— Si l'information est une expression textuelle : la variable aléatoire choisit sa valeur aléatoirement dans une liste de valeurs, stockées dans le corps de la classe si la variation est limitée (par exemple le type de contrat dans les fiches de paie françaises pourrait être CDD ou CDI) ou chargées à partir d'un fichier source si la variation est plus importante.

— Si l'information est une valeur numérique et qu'elle est calculée sur la base d'autres valeurs d'information, une fonction est utilisée pour la calculer, comme les totaux dans les factures et les tickets de caisse. Dans certains cas, les mots-clés et la valeur de l'information peuvent être vides aussi.

Les variations de l'information textuelle peuvent être enrichies en ajoutant de nouvelles valeurs aux listes des différentes classes, ainsi qu'aux fichiers de ressources.

La variation de la mise en page

Dans cette étape, nous créons et organisons des conteneurs qui contiennent le texte généré dans la première étape dans le DSS. Comme le montre la Figure 3.8, trois classes de mise en page mettent en œuvre une interface de mise en page générique (SSDLayout). Les trois classes définissent la fonction « buildModel » qui est responsable de la collecte et de la gestion de la mise en page du DSS et de la sauvegarde du DSS final au format PDF et au format image, ainsi que de l'annotation du DSS dans un fichier XML.

Les trois classes de mise en page (PayslipLayout, InvoiceLayout et ReceiptLayout) utilisent plusieurs composants de mise en page élémentaires dans la fonction « buildModel ». Ces composants représentent généralement les différents blocs de chaque type de DSS qui ont des configurations complexes. C'est le cas lorsque les blocs ont plusieurs groupes et ECs et qu'il existe différentes possibilités de les organiser par exemple : « EmployeeInfoBox » et « SalaryBox » dans les fiches de paie et « ProductBox » et « ReceiptProductBox » dans les tickets de caisse et les factures. Lorsque les blocs ont une composition simple (ils contiennent un groupe ou une paire), nous utilisons deux types de conteneurs génériques : « HorizontalContainer » et « VerticalContainer » pour les créer directement dans la fonction « BuidModel ».

Stratégie de variation de la mise en page Les éléments de mise en page du DSS (comme leur contenu) sont gérés à l'aide de différentes variables aléatoires décrites ci-après :

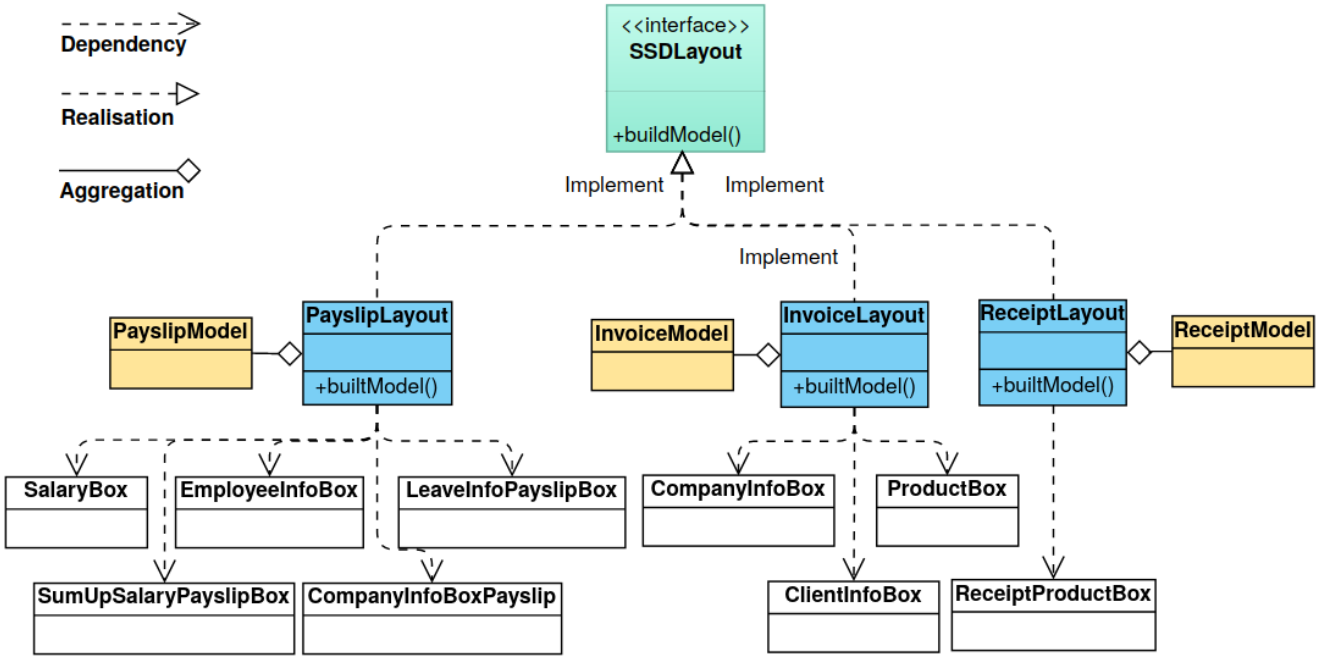


FIGURE 3.8 – Diagramme de classes UML de la génération d'un layout DSS. L'agrégation interprète les attributs de la classe Java ; la dépendance indique les classes utilisées dans la méthode buildModel ; et la réalisation représente l'implémentation de l'interface SSDLayout.

.

— pos_xi : choisit l'ordre des blocs (si un ou plusieurs blocs peuvent être placés à des endroits différents dans le DSS) ;

— $width\_random$: choisit la largeur et la hauteur du DSS à partir d'une liste de valeurs proposées ;

— xi_justif : sélectionne un alignement horizontal ou vertical parmi trois valeurs possibles pour chaque type d'alignement ;

— $columns\_order$: désigne l'ordre et la largeur des colonnes d'un tableau à partir d'une liste de valeurs fixes ;

— $font$: sélectionne une police de texte.

La Figure 3.9 montre l'algorithme de génération d'un exemple de DSS contenant 4 blocs obligatoires (A, B, C, D) et un bloc optionnel (E). Les blocs peuvent avoir deux ordres différents : A, B, C, D, E ou A, D, B, C, E. Chaque bloc contient un certain nombre de groupes et d'ECs obligatoires et facultatifs, comme le montre la Figure 3.9.

On commence par choisir la largeur du DSS et on crée son conteneur global. Le premier bloc A est généré ; la génération détaillée de ce dernier est la suivante : le bloc obligatoire A_1 est généré ; en fonction de la valeur de la variable aléatoire A2_available, nous décidons si le bloc optionnel A_2 sera généré ou non ; nous générons A_3, puis nous ajoutons tous les blocs générés au groupe G_{A1}. L'alignement du groupe G_{A1} est choisi en fonction de la valeur de la variable GA1_justif. Trois valeurs sont possibles : à droite (JR), à gauche (JL) et au centre (JC). Enfin, G_{A1} est ajouté à A, qui est à son tour ajouté au SSD_Container. En fonction de pos_b, la position du bloc B est choisie et, de la même manière, les blocs restants sont générés comme indiqué sur la Figure 3.9.

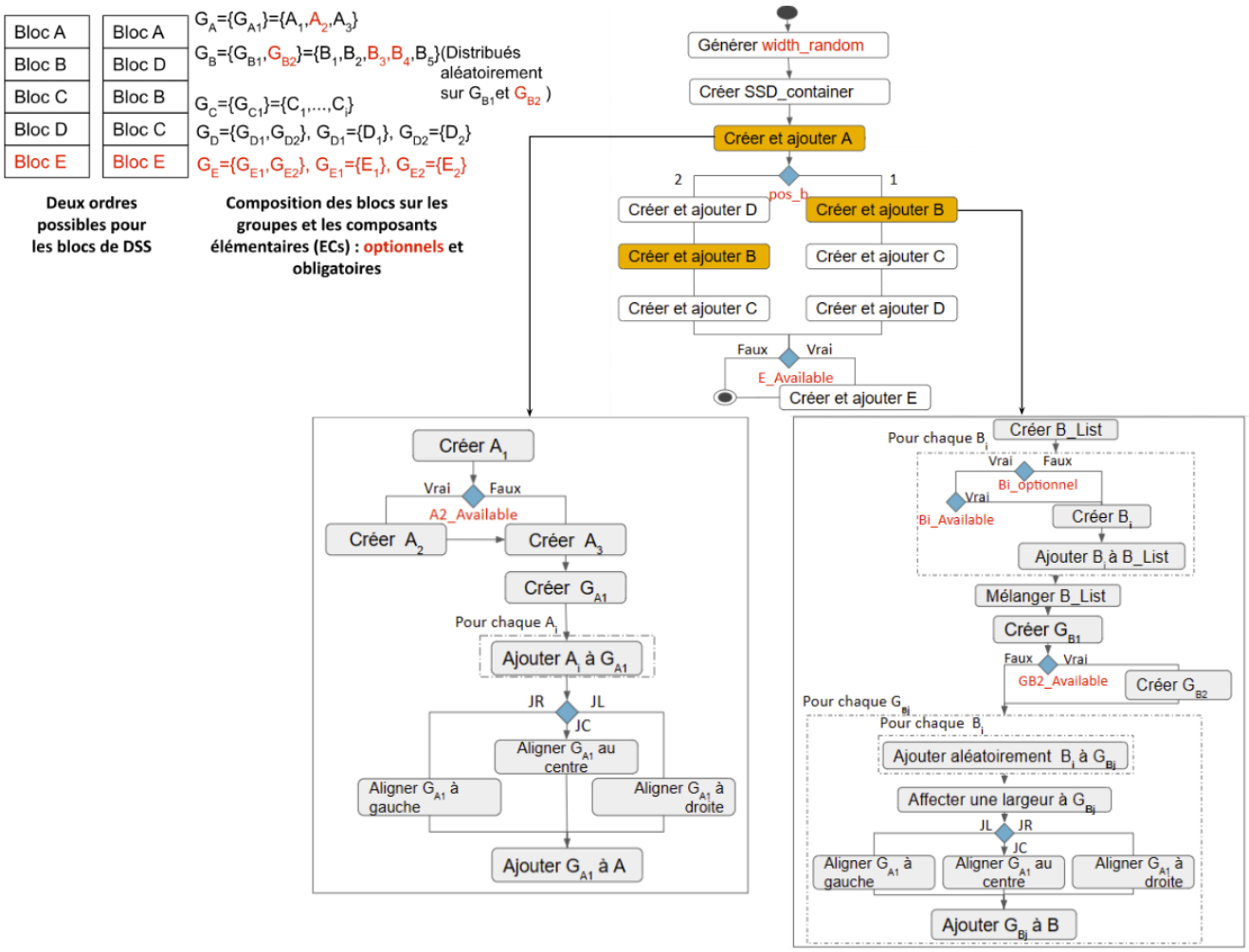


FIGURE 3.9 – Algorithme de génération des mises en page du DSS.

Pour ajouter de nouvelles variantes de mise en page aux trois types de DSS générés, nous ajoutons de nouvelles valeurs aux différentes listes de variables. Pour créer une nouvelle mise en page d'un nouveau type de DSS, nous ajoutons une classe de mise en page qui implémente l'interface SSDLayout, puis nous ajoutons de nouveaux blocs si nécessaire. Dans la méthode « buildModel », nous pouvons gérer ces différentes variations de mise en page.

3.2.6 Annotation

Les classes appropriées sont attribuées aux valeurs d'information et une classe « undefined » est donnée aux mots–clés et aux informations supplémentaires. La méthode « buildModel » enregistre chaque annotation de DSS dans un fichier XML. Ce dernier contient tous les attributs des mots du DSS : leurs classes, les coordonnées de leurs boîtes englobantes et les polices de texte. Une classe simpleTextBox qui rassemble toutes ces propriétés, est utilisée pour créer le composant de mise en page des étiquettes et des valeurs et enregistrer leurs propriétés dans le fichier XML.

3.2.7 Évaluation des jeux de données générés

Nous ajoutons une étape d'évaluation afin de choisir un jeu de données adéquat pour le processus d'apprentissage et d'éviter le sur-apprentissage. Le contenu du jeu de données générées doit être varié, de même que la mise en page, en fonction du type de DSS et de ses contraintes de génération.

La diversité du contenu textuel

Nous évaluons la diversité du contenu des jeux de données générés à l'aide d'une métrique très efficace qui permet d'évaluer la diversité des textes générés, connue sous le nom de métrique SELF-BLEU, proposée dans [153]. Elle mesure les différences entre les phrases générées de la même classe. Un score SELF-BLEU faible implique une grande variété. Le score BLEU, tel que défini dans [107], est calculé pour chaque phrase générée, et le score BLEU moyen est alors le SELF-BLEU de l'ensemble de données.

$$BLEU = PB * \exp(\sum_{n=1}^{N} w_n \log p_n) \tag{3.1}$$

où PB est la pénalité de brièveté. Elle est calculée comme suit :

$$PB = \begin{cases} 1 & \text{si } c > r \\ \exp(\frac{1-r}{c}) & \text{si } c \leq r \end{cases} \tag{3.2}$$

où c est la longueur de la phrase sélectionnée (l'hypothèse), r celle de la phrase de référence, N la longueur maximale du n-gramme et w_n un poids positif ($\frac{1}{N}$) et p_n le score de précision modifié qui est calculé comme suit,

$$p_n = \frac{\sum_{C \in Candidates} \sum_{n-gram \in C} Count_{clip}(n\text{-}gram)}{\sum_{C' \in Candidates} \sum_{n-gram \in C'} Count(n\text{-}gram)} \tag{3.3}$$

où $Candidates$ sont les phrases générées que nous voulons évaluer, $Count$ est le nombre de fois qu'un $n-gram$ apparaît dans la phrase candidate et $Count_{clip} = min(Count, Max\_Ref\_Count)$, où Max_Ref_Count est le nombre maximum de fois qu'un $n-gram$ apparaît dans une phrase de référence (les autres phrases générées).

La Table 3.3 montre le score SELF-BLEU pour trois jeux de données générés : tickets de caisse (G_Receipts), factures (G_Invoices) et fiches de paie (G_PaySlips). Chaque jeu de données contient 1000 DSSs. Le jeu de données SROIE est issu de la compétition ICDAR [60] et est pris ici comme référence. SROIE est un jeu de données publiques réelles qui contient 973 tickets de caisse. Nous remarquons que le score SELF-BLEU enregistré sur les trois jeux de données générés a une valeur faible, encore plus faible que le score enregistré sur SROIE qui est un jeu de données réelles. Cela signifie que les trois jeux de données générés présentent une grande diversité de contenu.

3.2.8 Diversité de la mise en page

La plupart des mesures utilisées pour calculer la diversité de la mise en page dans les documents générés calculent la distance entre les blocs de la même classe. Cette distance peut être donnée de différentes manières : la distance euclidienne entre les centres des blocs, les distances

TABLE 3.3 – Scores SELF-BLEU de le jeu de données SROIE et de nos trois jeux de données générés.

SROIE	G_Receipts	G_Invoices	G_Payslips
0.44	0.29	0.29	0.23

de Manhattan entre les coins (la distance Manhattan « d » entre deux points $a(x_1, y1)$ et $b(x_2, y_2)$ correspond à $d = |x_1 - x_2| + |y_1 - y_2|$), la différence entre la hauteur et la largeur, la zone de chevauchement entre les blocs ou des combinaisons de ces distances [124]. Un score d'alignement (Align) défini par la somme de la distance de Manhattan entre les deux coins des blocs nous donne des informations sur la position et la surface, tandis que la zone de chevauchement (Over) dépend de la position, de la taille et du rapport hauteur/largeur des blocs. Ces deux mesures pourraient servir d'indicateurs représentatifs et évidents des distances entre les blocs, car elles intègrent plusieurs informations sur leur disposition en une seule mesure.

Par conséquent, nous adoptons ces deux dernières mesures de diversité, introduites dans [124], pour calculer notre diversité d'alignement. Ces deux mesures (Align et Over) sont calculées sur les ECs obligatoires de chaque type de DSS. Nous définissons également une autre mesure que nous appelons SCR (SSD_Compositions Ratio) où SSD_composition d'un document est la liste de tous ses composants élémentaires (EC), qu'ils soient obligatoires ou facultatifs, ordonnés en fonction de leur position. Le SCR est une mesure de la diversité de la mise en page qui se concentre sur la composition générale de la mise en page du DSS.

Nous donnons dans ce qui suit les détails de ces trois mesures adoptées : Align, Over et SCR respectivement.

Soit $B_i = (p_a, p_b)$ et $B_j = (p_c, p_d)$ deux blocs EC obligatoires de la même classe dans deux mises en page de DSS, où p_a et p_b sont les coins qui définissent le bloc EC B_i (sa boîte englobante). Le score d'alignement ($Align$) est calculé comme suit :

$$Align(B_i, B_j) = Dmh(p_a, p_b) + Dmh(p_c, p_d) \tag{3.4}$$

où $Dmh(p_a, p_b) = |x_a - x_b| + |y_a - y_b|$ est la distance de Manhattan entre deux points p_a et p_b.

Le score de chevauchement $Over$ est calculé comme suit :

$$Over(B_i, B_j) = 1 - \frac{2 * ov(B_i, B_j)}{area(B_i) + area(B_j)} \tag{3.5}$$

où $ov(B_i, B_j)$ est la zone de chevauchement des deux blocs B_i et B_j, et $area(B_i)$ est la zone du bloc EC B_i.

Le SCR est calculé comme le nombre de $SSD\_compositions$ distinctes divisé par le nombre total de $SSD\_compositions$ qui est le même nombre de DSSs générés dans un jeu de données. Deux $SSD\_compositions$ sont considérées comme distinctes si leurs listes d'ECs ne sont pas parfaitement identiques (en comparant le nombre d'ECs, leurs classes ainsi que leur ordre dans la liste).

$$SCR = \frac{\#distinct\ SSD\_Compositions}{\#SSD} \tag{3.6}$$

L'alignement et les scores de chevauchement des blocs EC des DSSs sont effectués sur le jeu de données SROIE ainsi que sur 1000 échantillons générés de chaque type de DSS, comme indiqué dans la Table 3.4. Le score SCR est quant à lui calculé sur 2000 DSS générés de chaque type. Le SCR n'est pas calculé sur le jeu de données SROIE car le calcul est très coûteux en

Table 3.4 – Scores d'alignement, de chevauchement et de SCR.

Type de DSS	SROIE	G_Receipts	G_Invoices	G_Payslips
Align (alignement)	0.48	0.06	0.14	0.06
Over (chevauchement)	0.998	0.998	0.997	0.986
SCR	-	0.90	0.88	0.79

temps et nous n'avons pas assez d'informations sur toutes les classes d'ECs dans les échantillons SROIE.

Comme le montre la Table 3.4, le score d'alignement normalisé des trois jeux de données générées a des valeurs très faibles, même inférieures au score SROIE. La variation des positions des ECs dans les DSSs est la principale source de ces faibles valeurs et la variation du contenu affecte également ce score. Les coins des boîtes englobantes des ECs dépendent de la taille de la police du contenu des ECs et du nombre de caractères qui les composent.

Le score de chevauchement a des valeurs élevées, presque identiques à celles du score SROIE. Les restrictions concernant la position des ECs obligatoires dans les DSSs sont la première raison de cette valeur de score dans ce type de documents. La variation du contenu n'a pratiquement aucune influence sur ce score, même si les positions des coins des boîtes englobantes des ECs changent, mais leur zone partagée reste très importante. Et même pour les ECs qui ont des positions variables, la possibilité de chevauchement est très probable car la variation se situe généralement dans la même région du DSS.

Le score SCR, quant à lui, présente des valeurs élevées. Cela signifie que les configurations générées pour chaque type de DSS sont très diverses. Les différentes ECs optionnelles qui peuvent se trouver dans un DSS et pas dans un autre, ainsi que la variation de la position des ECs obligatoires et optionnelles, sont les principales explications de ce score de diversité élevé.

3.2.9 Synthèse

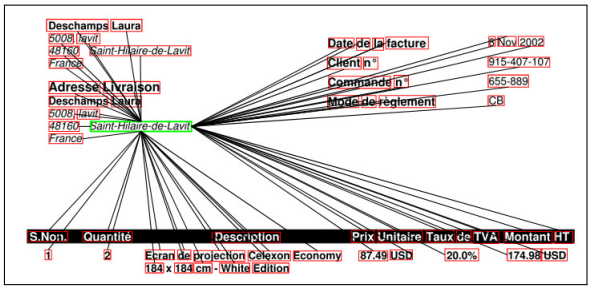
Dans cette section, nous avons présenté un générateur de documents semi-structurés (DSSs) qui génère automatiquement trois types de DSS (fiches de paie, tickets de caisse et factures) dont le contenu et la mise en page varient significativement. Sur la base de la structure générale d'un DSS, nous avons introduit des variables aléatoires, pour gérer les deux aspects (contenu et mise en page), qui garantissent la diversité du DSS. Nous avons fixé certaines contraintes pour nous assurer que nos documents artificiels soient relativement similaires aux documents réels. Nous avons ensuite introduit des métriques de mise en page et de contenu pour mesurer la diversité des DSSs générés. Les résultats de l'évaluation ont montré que les jeux de données générés pour les trois types de DSSs présentent une grande diversité de contenu et de mise en page. Cette diversité dans les jeux de données générés, évite le sur-apprentissage pendant le processus d'entraînement du système d'extraction d'informations.

3.3 Modèle d'EI transductif : $Multi - GAT_{Dataset}$

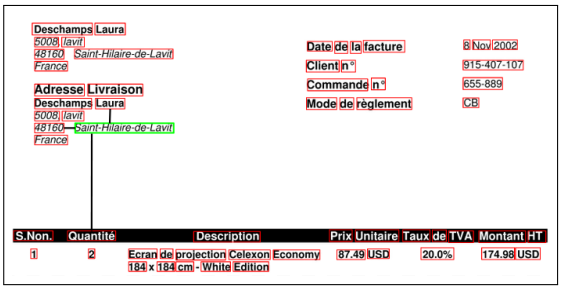
Dans cette section, nous présentons la première approche d'EI transductive « $Multi{-}GAT_{Dataset}$ ». Nous commençons par décrire la méthode de calcul du voisinage étoile dans le graphe, ainsi que le calcul des caractéristiques des mots. Nous détaillons ensuite l'architecture du Multi-GAT, le fonctionnement du mécanisme d'attention multi-têtes ainsi que l'approche d'apprentissage transductif.

3.3.1 Calcul du voisinage étoile dans le graphe

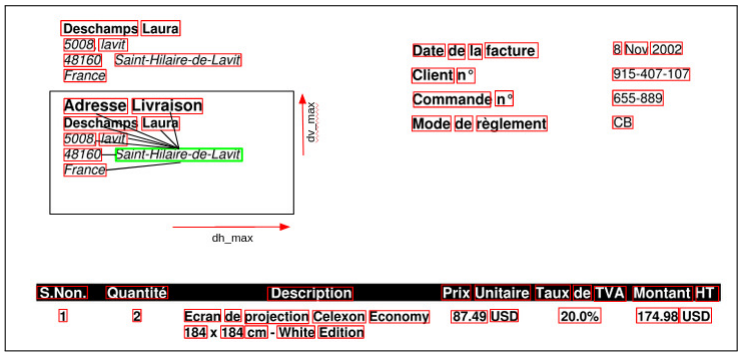
Le DSS est modélisé par un graphe de mots rassemblant des sous-graphes étoiles construits autour de chaque mot. Le voisinage d'un mot dans le DSS associe tous les mots situés à des distances horizontale et verticale limitées (dh_max et dv_max), formant ainsi un graphe étoile autour de chaque mot, comme illustré dans la Figure 3.10.c.



(a) Voisinage en graphe étoile [120].



(b) Le voisinage à quatre directions cardinales [94].



(c) Voisinage du sous-graphe étoile (le notre).

FIGURE 3.10 – Configurations de voisinages : a) Le mot est relié à tous les autres mots du document par un seul graphe étoile [120]. b) Le mot est connecté (au maximum) à ses 4 plus proches voisins dans les quatre directions principales (ici il est connecté à 3 autres mots) [94]. c) Notre proposition : le mot est connecté aux autres mots qui se trouvent à une distance limitée verticalement et horizontalement.

Pour définir ces distances, nous procédons d'abord à un étiquetage particulier de la base de données. À chaque entité à extraire est affectée un nom de classe. De plus, les mots-clés introduisant chaque classe d'information, reçoivent une nouvelle étiquette en ajoutant « K » au nom de la classe principale. Nous utilisons le nouvel étiquetage du DSS pour calculer les distances horizontales et verticales (dh et dv) séparant tous les mots de l'information de leurs mots-clés pour toutes les classes d'information. À la fin, nous prenons les distances maximales

les plus fréquentes enregistrées pour tous les mots appartenant aux différentes classes. À partir des distances dv_max et dh_max calculées à l'étape précédente, nous construisons un graphe non orienté $G = (V, E)$ ayant V comme ensemble de sommets et « E » comme ensemble d'arêtes. « V » contient les vecteurs de caractéristiques de tous les mots du DSS.

En comparaison avec deux autres approches qui modélisent un document comme un graphe, l'approche de [120] (voir Figure 3.10.a) connecte chaque mot à tous les autres mots du document qui, pour la plupart, sont à des distances très éloignées et ne contribuent pas à la classification des mots et alourdissent également le graphe. Par ailleurs, la proposition de [94] (voir Figure 3.10.b) prend un maximum de 4 voisins, ce qui peut exclure certains voisins importants qui pourraient aider à classer les mots dans le document. De plus, si les mots-clés ne sont pas suffisamment alignés avec les informations, ils peuvent ne pas être inclus dans ce voisinage. Notre proposition vise à rassembler le nombre maximum de voisins indicatifs et importants pour la classification des nœuds.

3.3.2 Calcul des caractéristiques des mots

Les mots dans le graphe sont décrits par leurs caractéristiques dans le document. Ces caractéristiques sont obtenues en combinant celles du texte avec celles de la mise en page, ainsi qu'une autre qu'on appelle « Nature du mot » et qui encode la catégorie du mot (alphabétique, numérique, etc.).

Pour former le vecteur de plongement textuel, nous utilisons les versions française et anglaise de BPEmb [54], un modèle basé sur BPE (Byte-Pair Encoding) [122]. Il s'est avéré être un outil efficace pour interpréter les images scannées et passées à un OCR. En combinant les sous-mots résultant de la segmentation, il peut déterminer plus précisément le sens du mot en dépit des erreurs de l'OCR. On a choisi d'utiliser un modèle non contextuel dans le but de créer un contexte adapté à notre modélisation du DSS par graphe. Les modèles pré-entraînés de BPEmb donnent néanmoins des représentations meilleures que celle du modèle non contextuel fastText par exemple, en nécessitant beaucoup moins de ressources de calcul (mémoire) [54]. Cette représentation permet de résoudre le problème des mots hors vocabulaire. Pour ces derniers, nous concaténons les représentations vectorielles des trois premiers sous-mots issus du processus de la segmentation du mot. L'utilisation d'un modèle pré-entraîné avec un vocabulaire de taille 200K et un vecteur de dimension 100, nous a permis d'obtenir un vecteur final de dimension 300 avec une précision importante. Si le nombre de sous-mots est inférieur à 3, on complète le vecteur par des paddings (0) jusqu'à obtenir la dimension 300 pour le vecteur final.

Le modèle que nous proposons, dans cette étape, est multilingue. Le même modèle est capable de traiter des documents en langue française et anglaise. Selon les résultats de la segmentation des mots dans le document, nous sélectionnons le modèle BPEmb à utiliser (BPE_Fr ou BPE_En). Nous utilisons l'Algorithme 1 pour générer le vecteur de plongement textuel de chaque mot dans le DSS.

Nous réutilisons également un sous-vecteur de dimension 8, représentant la nature du mot, extrait du vecteur binaire de base de dimension 13, suggéré par Lohani et al. [94] pour enrichir le plongement textuel. Ce vecteur (Nature du mot) indique si le mot appartient à l'une de ces catégories : alphabétique, numérique, alphaNumérique, nombre avec décimale, nombre réel, devise, nombre réel et devise, mélange (mot et devise). L'intérêt d'ajouter ce vecteur est de renforcer les caractéristiques textuelles.

Nous ajoutons à ces caractéristiques textuelles, les caractéristiques positionnelles du mot données par sa position normalisée dans le document. Il s'agit de deux doubles calculés comme suit : $x/largeur$ et $y/hauteur$, où x et y sont les coordonnées du coin le plus haut du rectangle

Algorithme 1 Calcul du vecteur des caractéristiques textuelles du mot w.

Entrée : w, BPE_En, BPE_Fr
Résultat : $Emb_{BPE}(w)$
  $Padding \leftarrow [0]_{100}$
  $S_w \leftarrow$ BPE(w) ▷ Segmenter w en un ensemble S_w de sous-mots en utilisant l'algorithme BPE
  Si $S_w[i]_{0 \leq i \leq |S_w|} \in BPE\_Fr$ **Alors :**
    $Emb \leftarrow Emb\_Fr$
  Sinon :
    $Emb \leftarrow Emb\_En$
  Fin Si
  Tant que $i \leq 3$ **Faire :**
    Si $S_w[i] \neq Null$ **Alors :**
      $Emb_{BPE} \leftarrow Emb_{BPE}||Emb(S_w[i])$
    Sinon :
      $Emb_{BPE} \leftarrow Emb_{BPE}||Padding$
    Fin Si
  Fin Tant que
  Retourner: Emb_{BPE}

englobant le mot dans le document, tandis que *largeur* et *hauteur* sont respectivement la largeur et la hauteur du document.

Nous obtenons finalement un vecteur de caractéristiques de dimension égale à 310 en concaténant ces trois parties comme suit : caractéristiques positionnelles || nature du mot || caractéristiques textuelles (Emb_{BPE}).

Les ensembles « V » et « E » composant les graphes des DSSs, sont modélisés respectivement par le biais des deux matrices : X (la matrice des caractéristiques) où X_{ij} représente la j ème caractéristique du i ème mot dans le graphe, et A (la matrice d'adjacence) où A_{ij} correspond à 1 si i et j sont voisins, 0 sinon.

3.3.3 Classifieur transductif $Multi - GAT_{Dataset}$

Afin de classer les noeuds du graphe construit, nous utilisons un réseau d'attention à graphe basé sur le mécanisme d'attention multi-têtes (Multi-GAT). Nous montrons en premier l'architecture retenue et le mode d'apprentissage utilisé. Ensuite, nous détaillons le fonctionnement du mécanisme d'attention multi-têtes.

Architecture transductive

Pour chaque type de DSS, tous les documents constituant le corpus d'apprentissage et de test, sont représentés par un graphe global déconnecté modélisé par deux matrices globales X et A. Cela est possible grâce à l'utilisation de la matrice A qui sépare les graphes unitaires des DSSs, comme on peut le voir dans la Figure 3.11. Ce graphe global est introduit dans un classifieur neuronal qui prédit les classes des nœuds.

Avec le type de voisinage proposé : les mots/nœuds indicatifs seront certainement inclus dans l'ensemble des voisins du nœud. L'ensemble des voisins peut également contenir d'autres mots qui ne sont pas utiles pour la classification des nœuds. Il existe donc des voisins plus importants que d'autres dans l'ensemble du voisinage. Nous ne pouvons pas savoir a priori quels sont les voisins

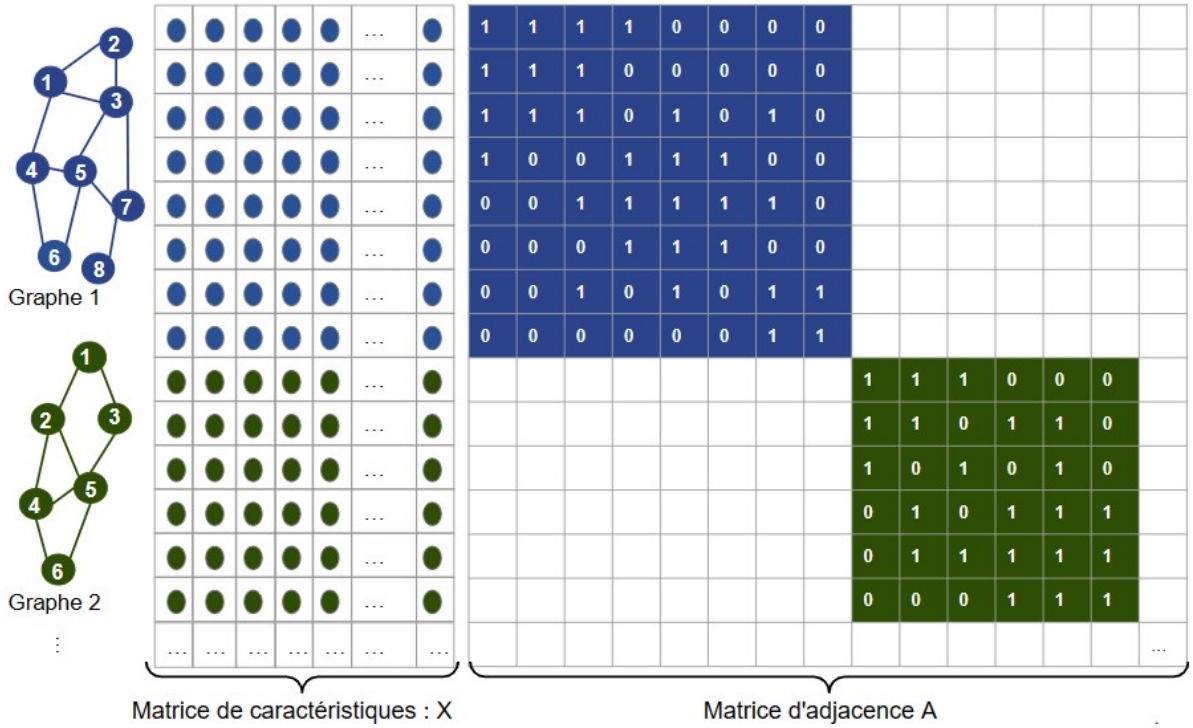


FIGURE 3.11 – La représentation des DSSs constituant un corpus par un graphe global déconnecté modélisé par deux matrices : la matrice de caractéristiques X et la matrice d'adjacence A. X contient les vecteurs de caractéristiques de tous les mots du corpus. Dans A, les graphes des DSSs apparaissent séparés grâce aux relations de voisinage calculées.

les plus ou les moins importants. Il est intéressant de choisir une méthode de convolution qui tienne compte de l'ensemble des voisins proposés et qui apprenne et attribue ensuite différents poids à ces voisins. Pour cela, nous avons proposé un classifieur utilisant le mécanisme d'attention multi-têtes. Ce dernier permet au réseau d'apprendre quels sont les voisins les plus importants et les moins importants pour chaque nœud du graphe. Cela permettra au modèle de se concentrer sur les nœuds voisins pertinents de chaque nœud du graphe, ce qui l'aidera à effectuer une meilleure classification des nœuds.

L'architecture du réseau neuronal est une architecture à trois couches GAT. Les deux premières appliquent l'attention multi-têtes avec k têtes (k varie en fonction du type de DSS et du nombre d'entités à extraire, comme nous le verrons plus loin) avec une fonction d'activation ReLU entre les couches. La couche de sortie applique une fonction Softmax pour classer les nœuds du graphe. Nous construisons l'architecture Multi-GAT, comme le montre la Figure 3.12, basée sur la couche GAT de base de [126].

L'apprentissage utilisé pour le modèle $Multi - GAT_{Dataset}$, dans cette première version du Multi-GAT, est l'apprentissage transductif. L'apprentissage du modèle final pour chaque type de DSS est fait sur tout le corpus. Le $Multi - GAT_{Dataset}$ peut donc voir au préalable, toutes les données, qu'il s'agisse de l'ensemble des données d'entraînement ou de celui des données de test. Afin de séparer les données d'apprentissage et de test, les classes des nœuds de test sont masquées lors du processus d'apprentissage du modèle. Cela permet de calculer la fonction objective uniquement sur les nœuds de l'entraînement (comme le montre la figure 3.12), en dépit du fait que le modèle puisse voir les valeurs des nœuds de test.

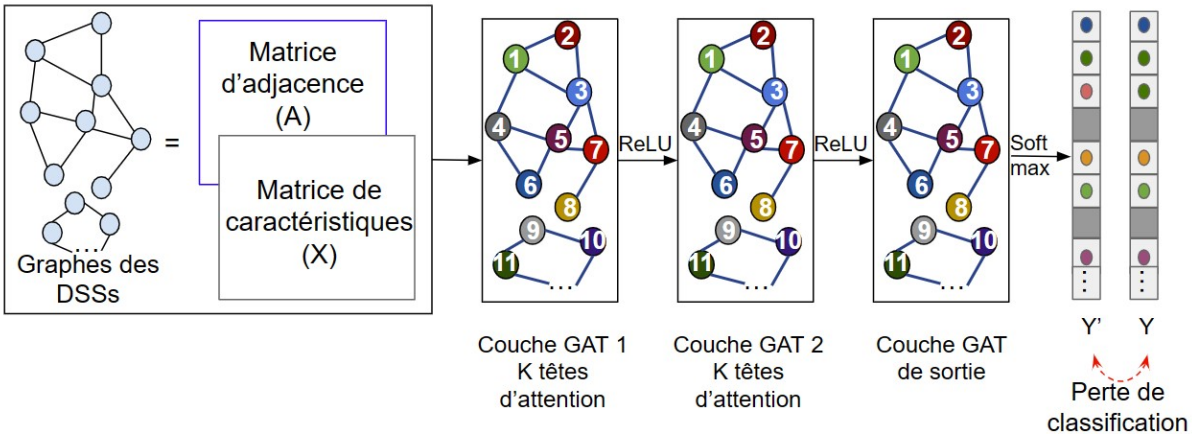


FIGURE 3.12 – Architecture transductive du $Multi - GAT_{Dataset}$.

Le mécanisme d'attention mutli-têtes

La convolution dans une couche GAT est calculée à l'aide du mécanisme d'attention multi-têtes illustré dans la Figure 3.13.

L'entrée du réseau est un ensemble de vecteurs de caractéristiques de tous les nœuds du graphe : $h = \{\overrightarrow{h_1}, \overrightarrow{h_2}, ...\overrightarrow{h_N}\}$ où $\overrightarrow{h_i} \in R^F$, où N est le nombre de nœuds et F est la taille du vecteur de caractéristiques de chaque nœud. F est égal à 310 dans l'entrée de la première couche. Cette couche génère un nouvel ensemble de caractéristiques de nœuds de taille F' (à l'exception de la couche de sortie, où on a habituellement $F' \geq F$), $h' = \{\overrightarrow{h'_1}, \overrightarrow{h'_2}, ...\overrightarrow{h'_N}\}$ où $\overrightarrow{h'_i} \in R^{F'}$.

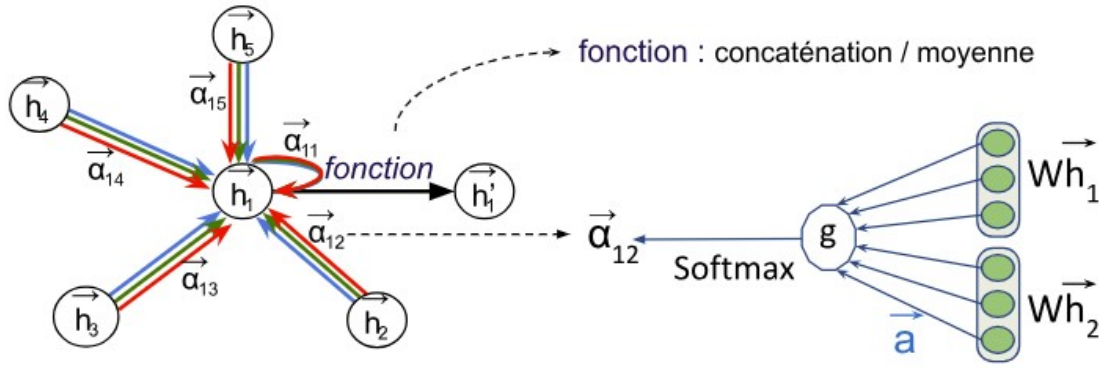


FIGURE 3.13 – Le mécanisme d'attention multi-têtes. Le nombre de têtes d'attention dans cet exemple est 3 et la fonction g correspond à $LeakyReLU$.

Les caractéristiques à la sortie de la couche GAT sont calculées comme suit : K mécanismes d'attention calculent K combinaisons linéaires des caractéristiques à l'aide des coefficients d'attention normalisés $\{\alpha_{ij}\}$ auxquels une fonction de non-linéarité σ est appliquée, puis fusionnent les résultats finaux, ce qui donne la représentation de l'Eq. 3.7.

$$\overrightarrow{h'}_i = \|_{k=1}^{K} \sigma\Big(\sum_{j \in N_i} \alpha_{ij}^k W^k \overrightarrow{h}_j\Big) \tag{3.7}$$

où $\|$ représente la concaténation, α_{ij}^k sont les coefficients d'attention normalisés calculés par le

k-ième mécanisme d'attention (a^k), et W^k est la matrice de poids de la transformation linéaire de l'entrée correspondante. La sortie finale renvoyée, h', se compose de caractéristiques de taille KF' (plutôt que F') pour chaque nœud.

Les coefficients calculés par le mécanisme d'attention sont les suivants :

$$\alpha_{ij} = Softmax(LeakyReLU(\overrightarrow{a}^T[W\overrightarrow{h}_i||W\overrightarrow{h}_j])) = \frac{exp(LeakyReLU(\overrightarrow{a}^T[W\overrightarrow{h}_i||W\overrightarrow{h}_j]))}{\sum_{k\in N_i} exp(LeakyReLU(\overrightarrow{a}^T[W\overrightarrow{h}_i||W\overrightarrow{h}_j]))} \tag{3.8}$$

Dans la couche de sortie du réseau (prédiction), une moyenne est utilisée à la place de la concaténation pour calculer les caractéristiques de sortie de la couche.

$$\overrightarrow{h'}_i = \sigma(\frac{1}{K}\sum_{k=1}^{K}\sum_{j\in N_i}\alpha_{ij}^k W^k \overrightarrow{h}_j) \tag{3.9}$$

Corrélation entre le nombre de têtes d'attention et le nombre de classes d'informations

Le mécanisme d'attention multi-têtes permet d'apprendre différents types de relations entre les nœuds du graphe. Chaque tête d'attention peut donc apprendre à se focaliser sur un type de relation particulier.

Le nombre de têtes peut dépendre de différents critères tels que la complexité de la tâche et du modèle, les ressources disponibles, etc. Un nombre plus important de têtes peut améliorer les performances du modèle, mais parallèlement, il entraîne une augmentation du temps d'apprentissage et d'espace mémoire. Afin de définir le nombre adéquat de têtes, nous pouvons utiliser la recherche aléatoire, en expérimentant différentes valeurs et en évaluant les performances du modèle sur un ensemble de tests. Enfin, nous choisissons le nombre qui optimise le mieux les performances du modèle.

Pour notre modèle, nous émettons l'hypothèse qu'il existe une corrélation entre le nombre de têtes d'attention dans le Multi-GAT et le nombre de classes d'information à extraire de chaque ensemble de DSSs. Cela signifie que chaque classe peut impliquer une relation (interprétable ou cachée) entre les nœuds, qui peut être apprise par une seule tête d'attention. Cela peut aider à orienter le choix du nombre de têtes d'attention pour chaque type de DSS. Cette hypothèse est vérifiée dans la section d'expérimentations.

3.3.4 Expérimentations et résultats

Dans cette section, nous présentons nos résultats en utilisant différentes métriques :

$Précision = \frac{VP}{VP+FP}$, $Rappel = \frac{VP}{VP+FN}$, $Accuracy = \frac{VP+VN}{VP+FP+FN+VN}$ et $F1 = \frac{2*Précision*Rappel}{Précision+Rappel}$

où : VP : Vrai positif, VN : Vrai négatif, FP : Faux positif et FN : Faux négatif.

Nous utilisons également l'entropie croisée catégorielle (ou perte Softmax) entre les étiquettes des noeuds et leurs prédictions pour calculer la perte de classification. Cette perte est définie par :

$$Perte = -\sum_x GT(x)\log P(x)$$

où GT est le vecteur des classes des noeuds « Ground Truth » et P est le résultat de prédiction des noeuds.

Le $Multi - GAT_{Dataset}$ est mis en œuvre sur la base de l'implémentation Keras tensorflow de la couche GAT proposée par [47]. Dans les différentes expérimentations, nous appliquons un nombre différent de têtes d'attention pour chaque type de DSS : 26 têtes d'attention pour les factures et les fiches de paie et 4, pour le jeu de données SROIE. Le facteur de régularisation L2 est fixé à 5.10^{-4}. Le taux d'apprentissage est fixé à 0.001 pour un maximum de 2000 époques et une patience de 100 pour l'arrêt anticipé.

Les jeux de données utilisés

C-Invoices : Ce jeu de données contient 1480 images de factures, correspondant à plus de sept modèles clonés (templates) réels en anglais et en français proposé dans Blanchard et al. [11], ce qui en fait un jeu de données à mise en page variable. Il comporte 26 classes de mots à reconnaître, comme on peut le voir dans la Table 3.6. Il est organisé comme suit : 70 % des noeuds du graphe pour l'entraînement, 20 % pour la validation et 10 % pour le test.

Gen-Payslips : Un ensemble de fiches de paie générées artificiellement par notre générateur ; il contient 1000 fiches de paie françaises. Chaque fiche de paie comporte 26 entités à extraire, comme montré dans la Table 3.7. Nous sélectionnons aléatoirement 70 % des noeuds du graphe global pour l'entraînement, 20 % pour la validation et 10 % pour le test.

SROIE : Un jeu de données de tickets de caisse qui est réuni dans la base de données publique SROIE [60], utilisée par la communauté de l'écrit. Elle contient 626 images de tickets de caisse pour l'entraînement et 347 images pour le test. Chaque ticket de caisse comporte quatre classes de mots à extraire : le nom de l'entreprise (Company), l'adresse (Address), la date (Date) et le total (Total). Pour correspondre à notre architecture, tous les autres mots n'appartenant pas à ces classes se voient attribuer la classe « undefined ». Ces tickets de caisse sont en anglais. SROIE est également un jeu de données avec une mise en page variable.

Les distances (dh_max, dv_max) calculées pour la génération du graphe pour les factures, les fiches de paie et les tickets de caisse, sont respectivement (900, 300), (1050, 200) et (800, 30).

Évaluation des composants du vecteur de caractéristiques

Afin de choisir le nombre de sous-mots à concaténer pour former les plongements Emb_{BPE} des mots hors vocabulaire, nous calculons le taux de sous-mots résultant de la segmentation de tous les mots dans SROIE, en utilisant les différents vocabulaires du BPEmb, comme on peut le voir dans la Figure 3.14.

Pour tous les vocabulaires, les mots sont segmentés en différents nombres de sous-mots (de 1 jusqu'à 14). En analysant les résultats de la segmentation, on remarque que pour les vocabulaires de petites tailles ($\leq 25K$), les sous-mots résultants sont peu significatifs (des caractères ou des 2-grammes/paires de caractères). En revanche, pour les vocabulaires de grandes tailles $\geq 50K$, on obtient une segmentation plus adaptée avec des sous-mots plus significatifs de la structure du mot (des préfixes et des suffixes, etc.). Par conséquent, nous sélectionnons le modèle BPEmb avec le vocabulaire de taille 200K car il présente le taux le plus faible des mots hors vocabulaire.

Comme montré dans la Figure 3.15, la majorité des mots hors vocabulaires (le nombre de sous-mots ≥ 1) dans le modèle BPEmb choisi, peuvent être segmentés en moins de quatre sous-mots. Les autres mots segmentés en quatre sous-mots ou plus, correspondent habituellement à

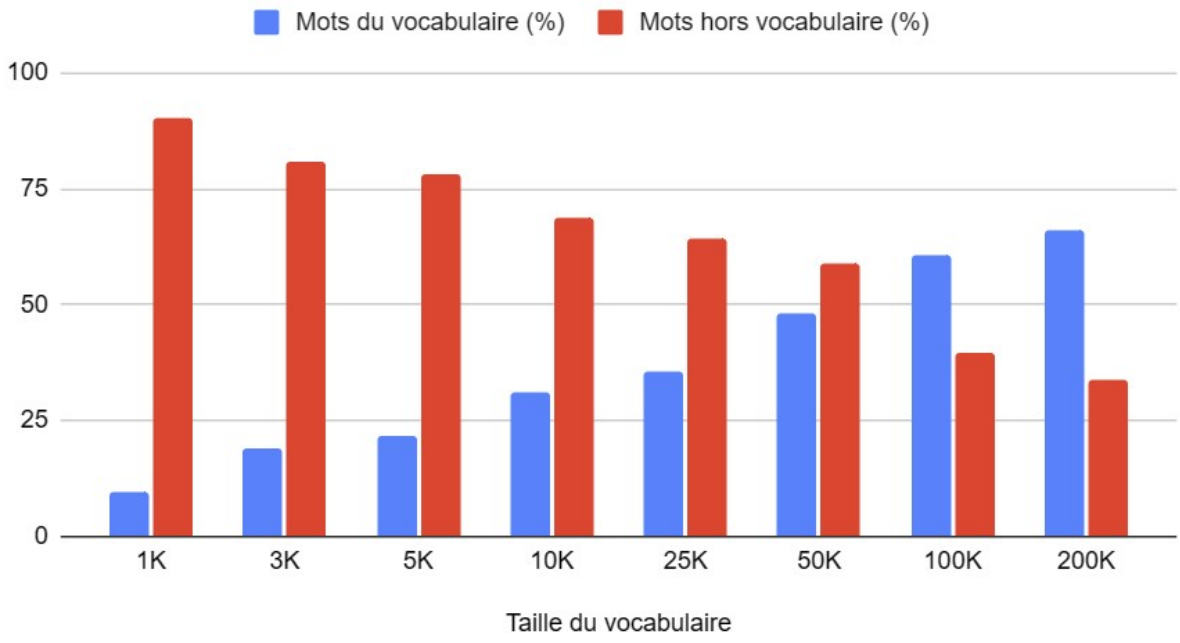


FIGURE 3.14 – Le taux (%) des sous-mots résultant de la segmentation des mots de SROIE en utilisant différents vocabulaires BPEmb.

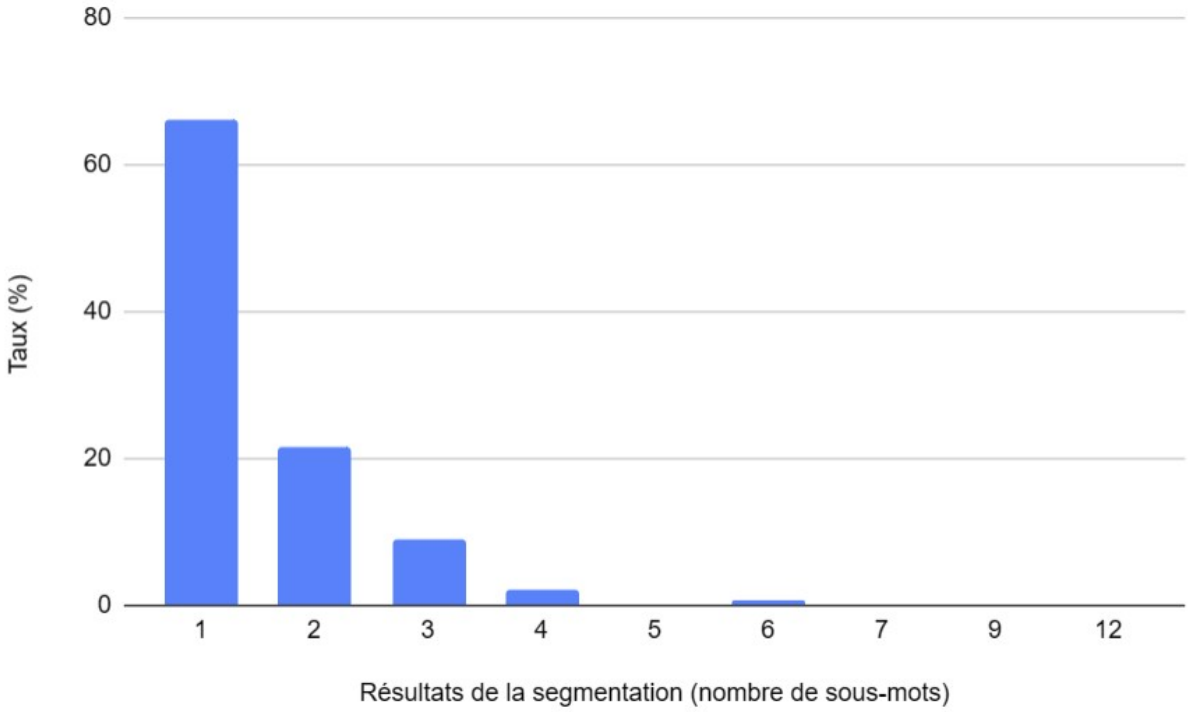


FIGURE 3.15 – Le taux (%) des nombres de sous-mots obtenus par segmentation avec le vocabulaire 200K sur SROIE. Le taux correspondant à un seul sous-mot représente le taux des mots appartenant au vocabulaire.

des url, des adresses web, etc. Ces mots sont généralement classés sous la classe "undefined", donc, la non prise en compte d'une partie du mot n'affecte pas les résultats du classifieur. Nous choisissons donc de prendre en considération uniquement les trois premiers sous-mots résultant de la segmentation.

Nous évaluons également l'effet des composants du vecteur de caractéristiques des mots dans le graphe sur les performances du classifieur, sur le jeu de données C-Invoices. Nous nous servons de la métrique de l'Accuracy en plus du score F1 pour mieux visualiser les différences de performance entre les configurations, étant donné que pour certaines d'entre elles, le score F1 est identique. Nous testons donc l'impact de chacune de ces caractéristiques : la position normalisée des mots dans le document (Pos. Norm.), la position absolue non normalisée (Pos.), le vecteur encodant la nature du mot (Nature (13)) proposé par [94], le vecteur de nature de dimension 8 (Nature (8)) que nous adoptons et l'effet des distances séparant chaque mot de ses voisins (Distances (Adj)) en utilisant une matrice d'adjacence pondérée par l'inverse des distances euclidiennes.

Caractéristiques des mots						Score	Accuracy
Emb_{BPE}	Pos. Norm.	Pos.	Nature (13)	Nature (8)	Distances (Adj)	F1 (%)	(%)
✓	✓	✗	✗	✗	✗	95	95.03
✓	✓	✗	✓	✗	✗	96	95.86
✓	✓	✗	✗	✓	✗	**97**	**96.82**
✓	✗	✓	✗	✓	✗	96	95.0
✓	✗	✓	✗	✓	✓	96	96.06

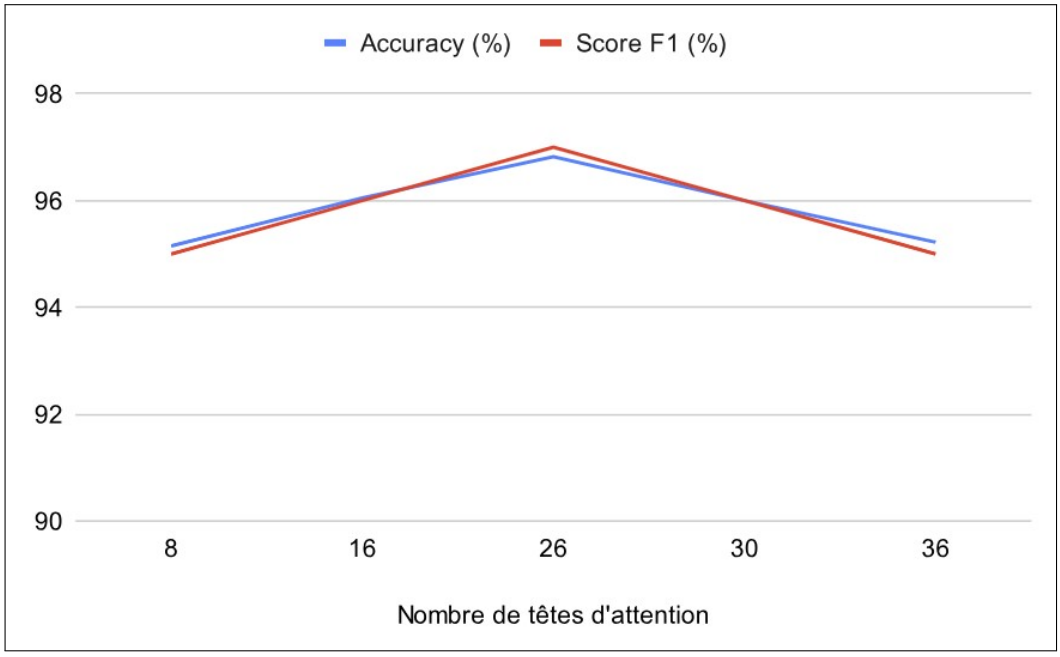
TABLE 3.5 – Évaluation de l'effet des composants du vecteur de caractéristiques sur le jeu de données C-Invoices. ✓indique la présence de la caractéristique dans le vecteur de caractéristiques final et ✗indique son absence.

Comme on peut le voir dans la Table 3.5, la position normalisée (Pos. Norm.) est plus efficace que la position non normalisée (Pos.). Nous notons également que les résultats du vecteur Nature (8) surpasse ceux du vecteur Nature (13). La différence entre eux est que Nature (13) calcule des booléens supplémentaires vérifiant si le mot appartient à une liste de noms de pays, département, etc. Cela entraîne un temps de calcul supplémentaire sans améliorer les performances du modèle. L'ajout des distances aux autres voisins dans la matrice d'adjacence n'améliore pas non plus les résultats. Cela peut bien signifier que le modèle apprend les distances relatives entre les mots à partir des positions absolues normalisées et l'utilisation de la matrice d'adjacence pondérée apporte une information redondante. On peut aussi déduire que les voisins les plus significatifs pour reconnaître la classe du mot peuvent ne pas être les plus proches (par la distance euclidienne) du mot.

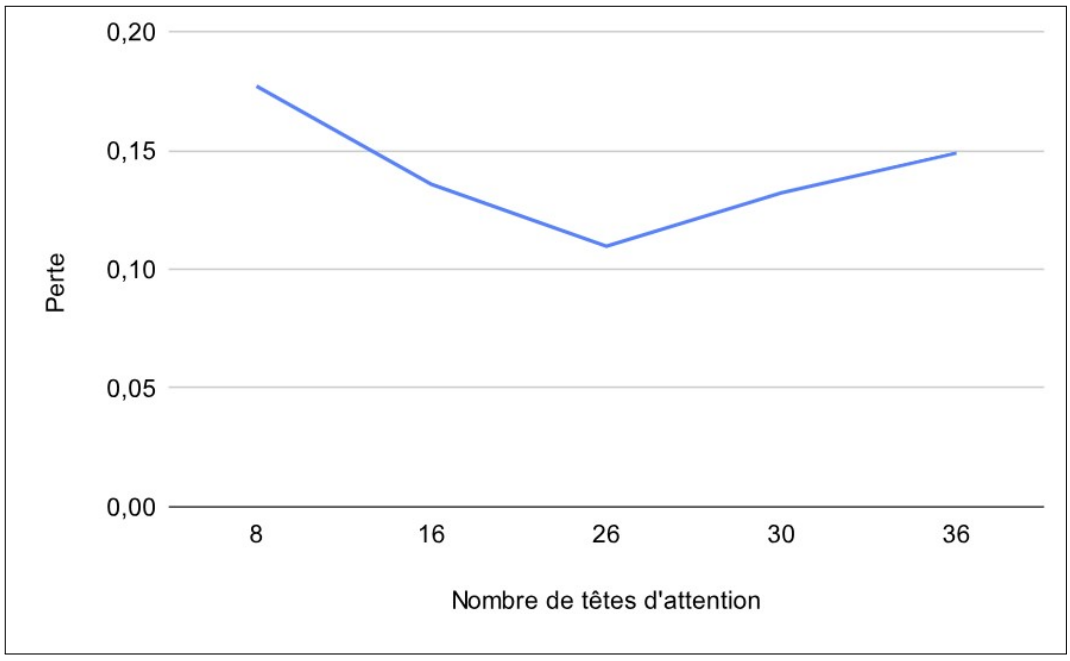
La structure finale du vecteur, formée par la concaténation du vecteur Nature (8), Pos. Norm. et le plongement du texte (Emb_{BPE}), donne le meilleur résultat.

Évaluation du nombre de têtes d'attention dans le GAT

Nous avons mené une étude pour analyser l'effet de variation du nombre de têtes d'attention sur les résultats d'extraction d'entités obtenus par le GAT. Comme nous pouvons le voir dans la Figure 3.16, le nombre de têtes d'attention 26 donne les meilleurs résultats sur le jeu de données C-Invoices avec : Perte = **0,1097**, Accuracy = **96.82 %** et F1 = **97%**. Les résultats diminuent en utilisant un nombre de têtes inférieur ou supérieur à 26, ce qui correspond au nombre de classes



(a) Accuracy (%) et score F1 (%) obtenus sur le jeu de données C-Invoices en variant le nombre de têtes d'attention.



(b) Perte obtenue sur le jeu de données C-Invoices en variant le nombre de têtes d'attention.

FIGURE 3.16 – Accuracy (%), score F1 (%) et la perte obtenus sur le jeu de données des factures (C-Invoices) en variant le nombre de têtes d'attention dans le mécanisme d'attention multi-têtes.

de nœuds. Nous supposons qu'il doit y avoir une corrélation entre le nombre de têtes d'attention et celui de classes de nœuds à prédire. Chaque classe peut intégrer un type de relation qui peut être appris par le réseau, qu'il soit interprétable ou caché.

Résultats détaillés sur les trois jeux de données

C-Invoices : La Table 3.6 montre les résultats du $Multi - GAT_{Dataset}$ comparés à ceux du système d'analyse de factures proposé par [94] sur le même jeu de données C-Invoices de 1480 factures clonées. Comme on peut le voir dans la Table, nos résultats sont meilleurs que ceux du GCN [94] sur la plupart des entités. Pour le « Total tax amount » et le « Total without tax », c'est le faible espacement entre eux et le fait qu'ils aient généralement les mêmes valeurs qui provoquent leur confusion.

TABLE 3.6 – Score F1 (%) sur le jeu de données C-Invoices en comparant le GCN (4 voisins) [94] et le $Multi - GAT_{Dataset}$.

Entité	GCN (4 voisins) [94]	$Multi - GAT_{Dataset}$
Invoice date	90	**99**
Invoice number	92	**98**
Order number	83	**91**
Payment mode	88	**100**
Company name	75	**87**
Company address	88	**94**
Company SIREN number	81	**93**
Company SIRET number	81	**92**
Company VAT number	91	**97**
Company APE code	85	**96**
Company Contact number	87	**96**
Company fax number	87	**97**
Client number	87	**90**
Client billing name	**90**	89
Client billing address	88	**90**
Client shipping name	**89**	88
Client shipping address	87	**90**
Product serial number	94	**99**
Product description	95	**98**
Product unit price	94	**100**
Product quantity	93	**97**
Product price without tax	**95**	91
Tax rate	99	**100**
Total without tax	**89**	77
Total tax amount	**93**	83
Net payable amount	**92**	91

Gen-Payslips : Les résultats présentés dans la Table 3.7 prouvent l'efficacité de notre système sur la base de données des fiches de paie. La plupart des classes ont un score supérieur à 91% (score F1). Différentes entreprises utilisent les mêmes mots-clés pour définir différentes informations détaillées et généralement facultatives qui peuvent entraîner une certaine confusion. « Employee qualification », « Coefficient » et « Earned leave » sont des exemples de ces entités, ce qui explique les scores obtenus pour elles. L'entité « Employee registration number » est également confondue avec l'entité « Employee qualification », ces deux entités ayant le même format, pouvant se trouver dans des positions similaires et les mots-clés qui les introduisent pouvant également se

ressembler. Le score F1 de l'entité « Company SIREN number » est de 57% car il est souvent confondu avec l'entité « Company SIRET number » et a des propriétés similaires. Dans les fiches de paie, on le trouve généralement sans mots-clés, ce qui augmente encore la confusion avec l'entité « Company SIRET number ».

TABLE 3.7 – Précision (%), Rappel (%) et score F1 (%) obtenus sur le jeu de données GenPayslips.

Classe	Précision	Rappel	Score F1
Employee name	99	94	96
Employment	92	96	94
Employee address	98	100	99
Employee code	92	96	94
Gross net pay	98	100	99
Social security number	100	100	100
Employee classification	95	92	94
Coefficient	76	83	79
Employee qualification	68	76	72
Starting date	89	93	91
Seniority	94	87	91
Payment date	100	98	99
Payment period	99	99	99
Company name	85	86	86
Earned leave	48	76	59
Company VAT number	93	95	94
Company address	95	94	94
Company APE code	94	85	89
Company SIRET number	73	93	82
Net salary	81	82	82
Company SIREN number	91	41	57
Net salary before taxes	80	83	81
Gross salary	100	100	100
Employee registration number	65	68	67
National collective agreement	98	95	97

SROIE : Nous avons comparé le score F1 moyen obtenu par le modèle $Multi-GAT_{Dataset}$ avec plusieurs autres approches basées sur le marquage des séquences de mots utilisant des modèles basés sur des RNN et BERT. Ces approches correspondent à l'architecture BiLSTM+CRF [61], un décodeur LSTM basé sur l'attention [81]. Les quatre meilleures méthodes de la tâche 3 de la compétition ICDAR [60], à savoir : le Top-4 qui correspond à la méthode « CLOVA OCR » qui utilise un modèle de marquage de séquence basé sur le modèle BERT [36] ; le Top-3 [97] qui propose un système d'étiquetage des séquences basé sur BiLSTM-CNNs-CRF et corrige ses résultats par des règles restrictives ; le Top-2 utilise le détecteur de texte [142] et un classifieur RNN ; Top-1 (RegEx) a été mis en œuvre par Ping An Property & Casualty Insurance Company. Il applique différents modèles d'expressions régulières pour extraire les informations clés.

Nous tenons à souligner qu'afin d'effectuer une comparaison plus fiable avec ces méthodes, nous avons utilisé la même annotation de SROIE proposée lors de la compétition [60], qui com-

porte certaines erreurs d'OCR et d'annotation.

TABLE 3.8 – Score F1 (%) sur le jeu de données SROIE en utilisant différentes approches basées sur les réseaux RNN, le modèle BERT et notre $Multi - GAT_{Dataset}$.

Méthode	Score F1 (%)
BiLSTM+CRF [61]	87.8
LSTM basé sur l'attention [81]	86.1
CLOVA OCR basé sur BERT [36]	89.05
BiLSTM-CNNs-CRF « H&H Lab » [97]	89.63
Détecteur de texte [142] + RNN	89.70
RegEx (Ping An Property & Casualty Insurance Company)	**90.49**
$Multi - GAT_{Dataset}$	89.2

Comme le montre la Table 3.8, nous avons obtenu un score F1 moyennement élevé (89.2%). Notre méthode surpasse toutes les méthodes basées sur les architectures RNN [61, 81, 97, 142] ainsi que la méthode CLOVA OCR basée sur BERT [36]. Certaines de ces méthodes sont spécifiquement conçues pour résoudre le problème de l'extraction d'entités à partir des tickets de caisse, comme RegEx qui a été classée première lors de la compétition d'ICDAR [60]. Cette méthode identifie les 4 entités avec des expressions régulières adaptées, ce qui la rend applicable uniquement sur ce jeu de données et restreinte aux quatre entités prédéfinies. Notre modèle diffère de cette approche qui s'applique uniquement aux tickets de caisse. Il est générique et peut être utilisé pour tout type de DSS, quel que soit le nombre de classes d'information à extraire.

En outre, contrairement à [97] qui utilise un post-traitement afin de corriger et améliorer les résultats en fixant certaines contraintes prédéterminées, nous ne présentons que les résultats immédiats de notre modèle (sans post-traitement).

Cette évaluation fait état d'erreurs provenant de sources multiples. Il s'agit notamment d'erreurs d'annotation qui nuisent au processus d'apprentissage, en particulier en ce qui concerne l'entité « Total ». En outre, les erreurs d'OCR ont également un impact négatif sur les performances du modèle. Des erreurs de confusion ont aussi été observées entre les mots appartenant aux entités « Company » et « Address ». En effet, ces deux entités ne sont pas introduites par des mots-clés et se trouvent dans les mêmes emplacements sous forme de deux informations mulilignes successives qui ne sont généralement pas séparées. Cela peut rendre la distinction entre les deux entités plus difficile à établir surtout lorsque les deux entités ont un un contenu semblable.

3.3.5 Synthèse

Nous avons prouvé l'efficacité de cette première version du $Multi - GAT_{Dataset}$ sur les trois types de DSSs utilisés. Ce modèle surpasse le GCN [94] sur la base de données C-Invoices et plusieurs architectures basées sur les architectures RNN et BERT pour l'étiquetage des séquences du texte sur SROIE. Nous avons également prouvé qu'une corrélation peut exister entre le nombre de têtes d'attention dans le Multi-GAT et le nombre de classes d'informations à extraire. Le Multi-GAT a la capacité de se concentrer sur les voisins des nœuds les plus importants en utilisant le mécanisme d'attention et donc de mieux prédire leur classes.

En revanche, le calcul du voisinage dans le graphe nécessite une étape de prétraitement sur un jeu de données connues, ce qui peut conduire à une difficulté pour généraliser les distances calculées sur de nouveaux documents avec des mises en pages différentes. De plus, si les documents d'un type particulier de DSS présentent des distances dh et dv très dispersées entre les voisins,

la sélection des distances les plus récurrentes correspondant à dh_max et dv_max devient plus difficile. En outre, l'approche d'apprentissage transductif peut améliorer les performances en exploitant les données de test, mais elle ne permet pas une généralisation naturelle aux nœuds non vus. Elle ne permet pas de déduire comment le modèle se comporte avec des données jamais vues auparavant.

Pour rendre ce classifieur plus générique, plus efficace et moins complexe, nous proposons une version améliorée du calcul du graphe et de l'architecture du Multi-GAT également.

3.4 Modèle d'EI inductif : $Multi - GAT_{DSS}$

Afin de rendre le Multi-GAT multimodal, plus générique aux nouvelles données et moins coûteux en mémoire, nous proposons une approche inductive d'EI en utilisant un Multi-GAT multimodal inductif. Nous allons exposer dans la suite, les améliorations que nous avons apportées à la modélisation du DSS, à l'architecture et à l'entraînement du Multi-GAT.

Nous proposons une nouvelle méthode plus générique de calcul des voisins des mots dans le DSS, qui peut être adaptée à n'importe quel type de DSS sans qu'une étape de prétraitement ne soit nécessaire. Cette méthode sélectionne un ensemble restreint de voisins les plus proches du mot dans le DSS. Nous limitons ce voisinage à la ligne du mot et aux deux lignes qui l'entourent. Nous enrichissons également les caractéristiques mutlimodales des noeuds en ajoutant les caractéristiques visuelles et l'encodage de leur région dans l'image. La matrice d'adjacence est contrainte à une taille unique. Elle peut inclure un seul DSS, une partie du DSS ou plusieurs DSSs. Le modèle Multi-GAT noté « $Multi - GAT_{DSS}$ » est ainsi entraîné en mode inductif où il ne fait usage que de l'ensemble des données d'entraînement, contrairement au modèle transductif « $Multi - GAT_{Dataset}$ » qui est entraîné en ayant une visibilité à la fois sur les données d'entraînement et sur les données de test.

3.4.1 Nouveau calcul du voisinage dans le graphe

Le calcul du voisinage étoile présente plusieurs limites. D'abord, on peut citer la complexité du choix des distances maximales, en particulier lorsqu'il s'agit de documents dont la mise en page est très variée. Ensuite, la présence de mots de tailles et de polices différentes peut entraîner des distances instables et variables. Un prétraitement doit donc être fait pour définir les distances maximales pour chaque type de DSS. Pour palier ces limites, nous proposons une nouvelle méthode de calcul du voisinage qui prend en compte les différentes caractéristiques des DSSs. Comme on a pu le remarquer dans la section 3.2, l'emplacement des mots-clés par rapport

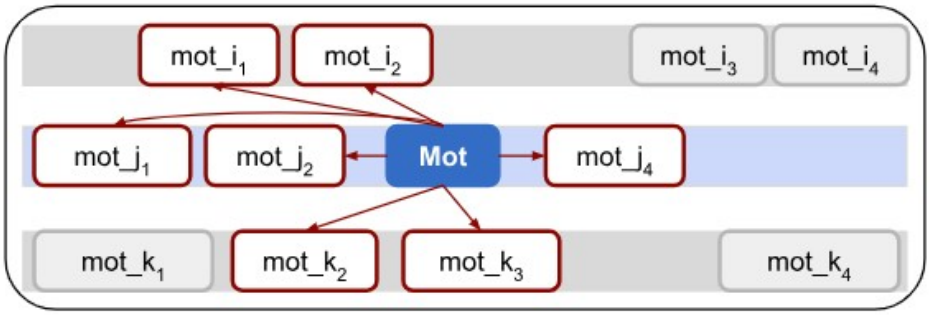
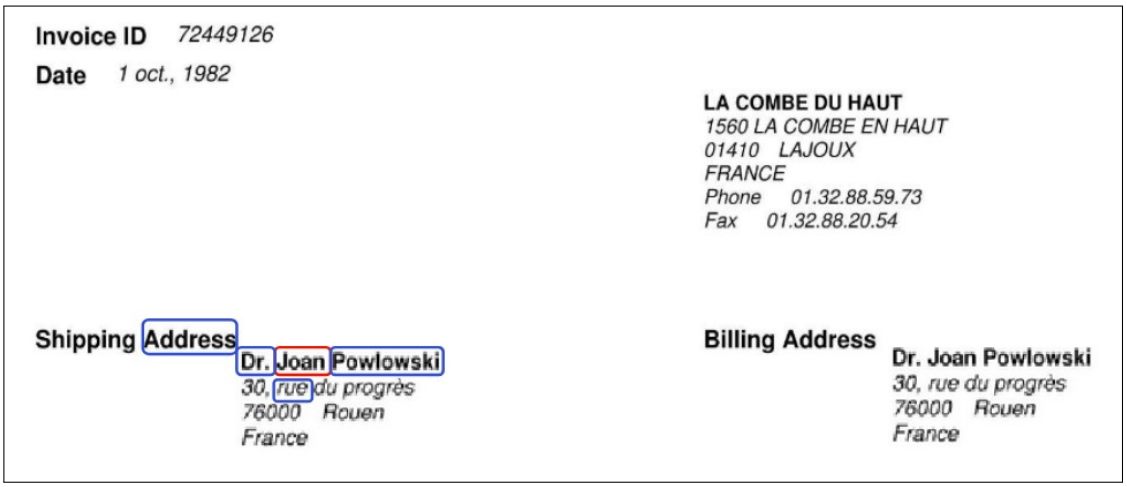


FIGURE 3.17 – Sélection du voisinage du mot dans le graphe.

aux informations dans le DSS se présente sous plusieurs configurations. Les mots-clés peuvent être alignés horizontalement avec l'information (sur la même ligne, à gauche), partiellement ou complètement alignés verticalement, ou placés sur la ligne au-dessus sans alignement mais toujours à proximité spatiale de l'information. Pour chaque mot appartenant à une information, tous les autres mots faisant partie de la même information ou de ses mots-clés introductifs, peuvent contribuer à reconnaître la classe du mot.

Pour chaque mot dans le DSS, nous sélectionnons les « k » voisins les plus proches situés sur sa ligne ainsi que ceux sur la ligne au-dessus et celle au-dessous, si elles existent, comme montré dans la sélection de voisinage de la Figure 3.17. Pour cela, nous utilisons la distance euclidienne entre les boîtes englobantes. Le nombre total maximum de mots voisins est de $n = 4 * k$ où k est fixé expérimentalement. Le voisinage sélectionné est supposé contenir les mots-clés indicatifs qui introduisent les informations que nous voulons extraire du DSS.

(a) Exemple du calcul du voisinage à quatre voisins [94].



(b) Exemple du calcul du voisinage sur trois lignes.

FIGURE 3.18 – Comparaison entre le calcul du voisinage à quatre voisins [94] et notre proposition sur trois lignes du mot « Joan » (entouré en rouge). (a) Les voisins sélectionnés avec la méthode des 4 voisins, sont entourés en vert. (b) Les voisins sélectionnés avec la nouvelle méthode de voisinage (en prenant k=1), sont entourés en bleu.

Comme on peut le voir dans la Figure 3.18, la méthode des quatre voisins, proposée par [94], risque de prendre des mots très éloignés du mot en question comme des voisins immédiats (« 72449126 » est un voisin immédiat du mot « Joan »). Ces voisins spatialement éloignés habituellement, n'apportent aucune information utile pour la prédiction des classes des entités dans les DSSs et peuvent même nuire à leur bonne classification en éliminant les voisins les plus indicatifs (dans cet exemple « Address » ou « Shipping Address »). Contrairement à cette méthode, notre approche réussit mieux à sélectionner les voisins les plus utiles à la classification des mots.

3.4.2 Caractéristiques multimodales des mots.

Nous proposons d'enrichir le vecteur de caractéristiques des mots en le complétant par d'autres aspects multimodaux (voir Figure 3.19). Le vecteur final contient les caractéristiques suivantes :

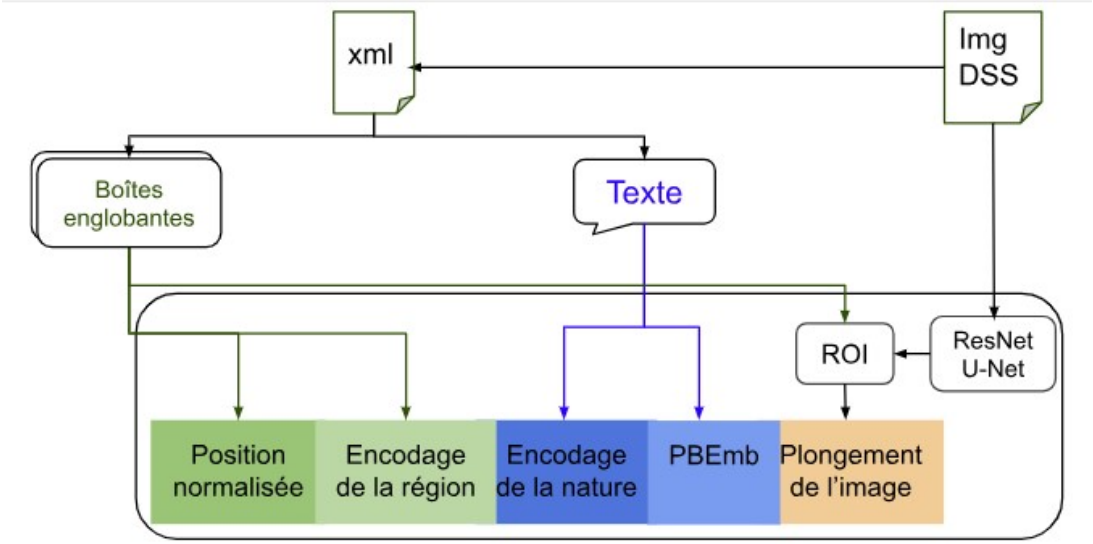


FIGURE 3.19 – Structure du vecteur de caractéristiques multimodales.

— Caractéristiques textuelles : nous réutilisons les mêmes caractéristiques textuelles que dans la première version du modèle $Multi - GAT_{Dataset}$, à savoir la concaténation de la représentation des trois premiers sous-mots, obtenue à l'aide de BPEmb [54] et du vecteur booléen « Nature (8) » de dimension 8, représentant la nature du mot (alphabétique, alphanumérique, etc.).

— Caractéristiques positionnelles (position normalisée) : au lieu de normaliser la position sur la largeur et la hauteur de l'image complète, nous proposons une nouvelle normalisation de la position absolue relative à la zone contenant le texte seulement. Comme on peut le voir dans la Figure 3.20a, cette zone peut varier d'un document à l'autre dans les images scannées. Tout d'abord, la boite englobante de toute la région entourant le texte ayant les coordonnées (x_0, y_0, x_n, y_n) est extraite. Le point (x_0, y_0) devient alors l'origine du nouveau repère que nous allons utiliser pour le calcul de la position normalisée et l'encodage de la région. Les nouvelles coordonnées des mots sont calculées en fonction de ce nouveau repère. Nous choisissons le point supérieur gauche et le point inférieur droit de la boîte englobante et nous calculons leurs coordonnées normalisées dans le nouveau référentiel, comme on peut le voir dans la Figure 3.20b.

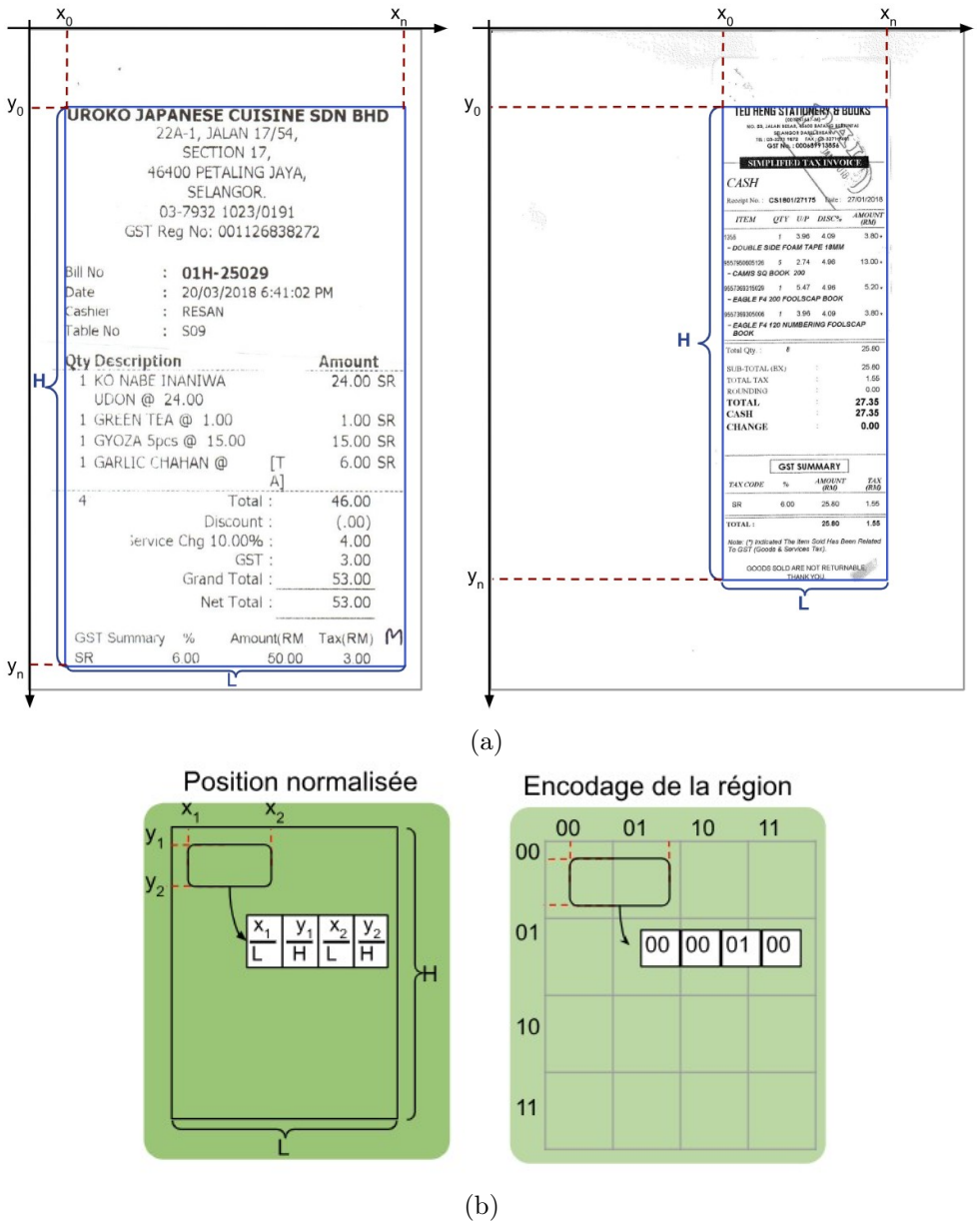


FIGURE 3.20 – (a) Exemples de la largeur et la hauteur des documents relatives à la région entourant le texte et (b) Calcul de la position normalisée et l'encodage de la région dans le nouveau repère.

— Encodage de la région : nous avons remarqué que les informations ont une position variable d'un document à l'autre, mais qu'elles se retrouvent généralement dans la même partie du document, c'est-à-dire dans une région spécifique. Pour exploiter cette particularité, un nouvel encodage de région est ajouté. La zone entourant le texte dans le document est divisée horizontalement et verticalement par 4, donnant lieu à 16 régions de la page. Chaque région est ensuite représentée par un vecteur binaire de dimension quatre : deux bits $(00, 01, 10, 11)$ encodant le numéro de la région sur l'axe des abscisses et deux autres encodant le numéro de la région selon l'axe des ordonnées. Chaque boîte englobante est

ensuite représentée par un vecteur de dimension 8 regroupant l'encodage de la région contenant le point supérieur gauche ainsi que la région contenant le point inférieur droit.

— Caractéristiques visuelles : un vecteur de plongement d'image est calculé pour chaque mot du DSS. Le réseau ResNet Unet [53, 119] est utilisé pour cela. L'image d'entrée est redimensionnée pour unifier la dimension d'entrée du modèle, puis elle est transmise à un encodeur ResNet Unet pré-entraîné. En utilisant la carte des caractéristiques résultante et les coordonnées des boîtes englobantes des mots, nous extrayons pour chaque mot sa région d'intérêt (Region Of Interest (ROI)). Enfin, nous transformons le résultat de la région d'intérêt en un vecteur 1D de taille 21, comme on peut le voir dans la Figure 3.21.

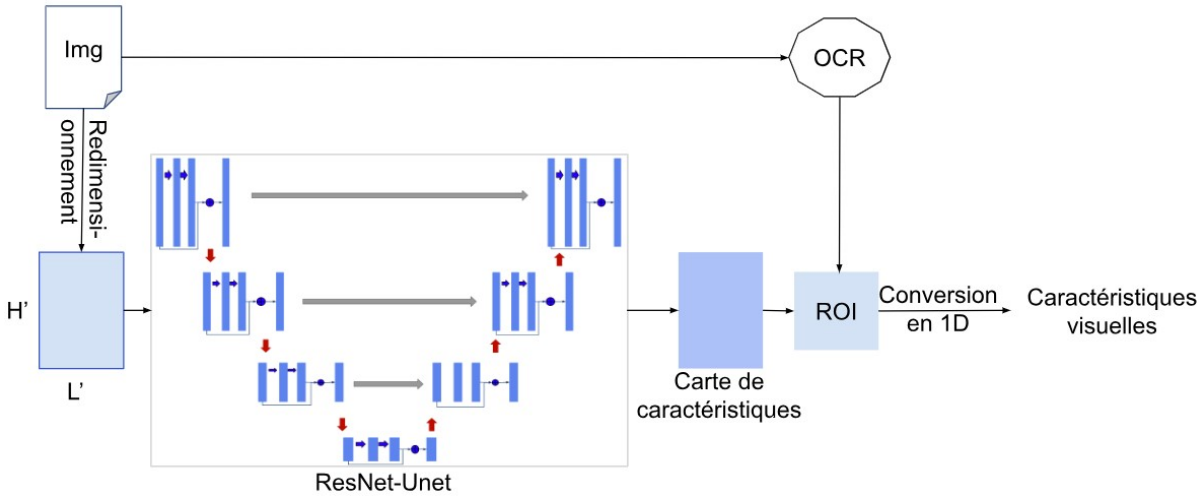


FIGURE 3.21 – Calcul des caractéristiques visuelles.

Le vecteur de caractéristiques global est une concaténation de toutes ces caractéristiques multimodales, formant un vecteur de taille 337, comme montré dans la Figure 3.19.

3.4.3 Algorithme de construction des matrices X et A

Le système précédent utilise un graphe global de tous les mots du jeu de données d'apprentissage, ce qui le rend très complexe et augmente le temps de traitement des DSS.

Nous proposons ici de limiter les graphes à un maximum de n_max=256 nœuds (256 est la puissance de 2 la plus proche du nombre moyen de mots dans tous les graphes de DSSs étudiés ici). Cela se fait soit en associant plusieurs graphes de moins de 256 nœuds dans le même graphe, soit en divisant un graphe de plus de 256 nœuds en plusieurs graphes de 256 nœuds maximum chacun, comme le montre la Figure 3.22. Cela est rendu possible grâce au fait que le voisinage des nœuds dans le graphe est limité, ce qui permet de dissocier ces nœuds les uns des autres.

L'algorithme utilisé pour former les matrices X et A pendant l'apprentissage, est détaillé dans l'Algorithme 2. Chaque tuplet de matrices (X, A, Y) est rempli progressivement par les noeuds et leurs voisins dans chaque DSS jusqu'à atteindre la taille maximale. On sauvegarde le tuplet lorsque l'ajout du noeud et ses voisins entraîne le dépassement de n_max. Quand cela arrive, on complète les matrices par des paddings (0) et on les ajoute à $liste_{DSSs}$, puis on initialise un nouveau tuplet en ajoutant le noeud en question et ses voisins à de nouvelles matrices. On continue cette procédure jusqu'à modéliser tous les documents dans le jeu de données par la liste $liste_{DSSs}$.

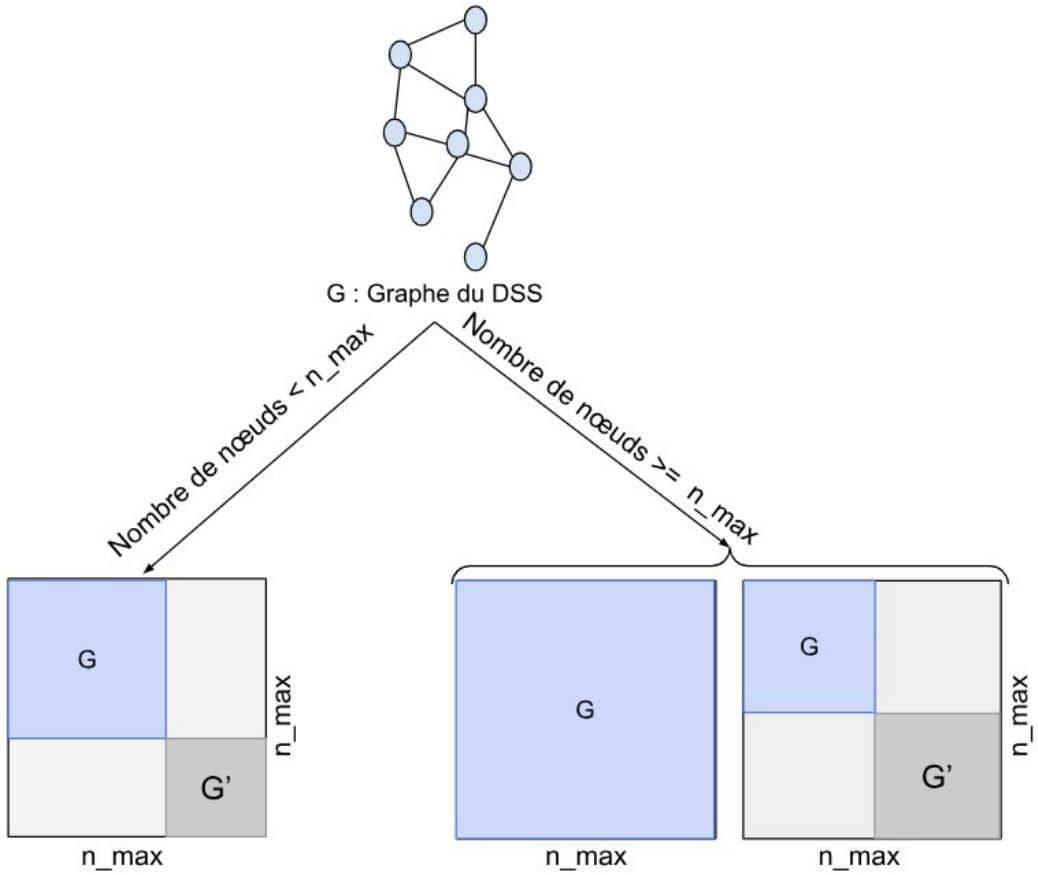


FIGURE 3.22 – Construction de la matrice d'adjacence du graphe G. Si le nombre de nœuds dans G est inférieur à n_max, il est représenté par une seule matrice d'adjacence, sinon il est séparé en plusieurs matrices de dimension n_max, en veillant à inclure tous les voisins immédiats de chaque nœud.

3.4.4 Classifieur inductif $Multi - GAT_{DSS}$

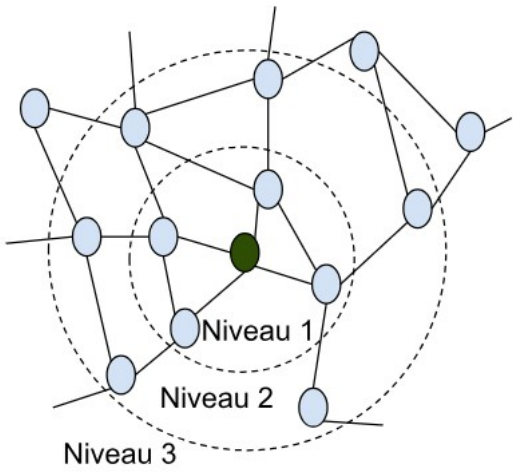
Afin de classer les noeuds du graphe construit, nous modifions légèrement l'architecture du classifieur $Multi - GAT_{Dataset}$ proposée dans la section précédente. Pour chaque type de document (facture, fiche de paie ou ticket de caisse), un nombre différent de têtes d'attention « k » est fixé en fonction de l'hypothèse proposée dans la première version du Multi-GAT concernant la corrélation entre le nombre de têtes d'attention et le nombre d'entités à extraire du DSS.

Un modèle de base contenant quatre couches, est adopté pour tous les types de DSSs. De plus, un modèle large contenant six couches, est ajouté pour les DSSs de type tickets de caisse. Le nombre de couches pour chaque DSS est fixé expérimentalement, en le variant et en évaluant les performances du modèle. Chaque couche du modèle prend en considération un voisinage d'un niveau plus lointain des mots dans le graphe : la convolution dans chaque couche GAT permet de prendre des voisins de plus en plus éloignés, comme on peut le voir dans la Figure 3.23b. Le voisinage global obtenu englobe les informations appartenant au même bloc d'information du mot. L'ordre des informations dans chaque bloc d'information dans le DSS peut être utile pour prédire les classes des mots. Par exemple, les lignes qui précèdent l'entité « Total » dans un ticket de caisse, contiennent souvent les informations de « Total avant taxe », « Total après

Algorithme 2 : Construction des matrices d'adjacence et de caractéristiques pour le $Multi - GAT_{DSS}$.

Entrée : $liste_{DSSs}, n\_max$
Résultat : $liste(X, A, Y)$
  $X, A, Y \leftarrow$ Null
  Pour chaque DSS dans $liste_{DSSs}$ **Faire :**
    $X_0 \leftarrow$ Calcul des caractéristiques (DSS)
    $A_0 \leftarrow$ Calcul du voisinage (DSS)
    $Y_0 \leftarrow$ Classes des mots du DSS
    Si $Taille(A_0) \leq n\_max - Taille(A)$ **Alors :**
      $X \leftarrow X_0$, $A \leftarrow A_0$, $Y \leftarrow Y_0$
    Sinon :
      Pour chaque noeud i dans A_0 **Faire :**
        Si Taille(A_{tmp}+ i + Voisins(i)) $\leq A_{tmp}$ **Alors :**
          Ajouter i et les voisins directs de i à A_{tmp}
          Ajouter i et les voisins à X_{tmp}
          Ajouter i à Y_{tmp}
        Sinon :
          Ajouter $(X_{tmp}, A_{tmp}, Y_{tmp})$ à (X, A, Y)
          Ajouter des Paddings (0) à (X, A, Y)
          Ajouter (X,A,Y) à $liste_{DSSs}$
        Fin Si
      Fin Pour
    Fin Si
  Fin Pour
  Retourner: $liste_{DSSs}$

(a) Bloc d'information de l'entité « Total ».



(b) Les niveaux du voisinage dans le graphe.

FIGURE 3.23 – (a) Exemple du bloc d'information de l'entité « Total » dans un ticket de caisse. (b) les niveaux de voisinage dans le graphe du noeud vert.

taxe », « Remise », « Montant arrondi », etc. et les lignes qui suivent cette entité, peuvent aussi

contenir les informations sur le montant et le moyen de paiement (voir la Figure 3.23).

Pendant l'étape d'apprentissage, le corpus des DSSs d'entraînement est représenté par des graphes modélisés par un ensemble de matrices X et A, construites comme expliqué ci-dessus. Chaque couple (X, A) correspond à un seul batch d'entraînement. Le $Multi - GAT_{DSS}$ est entraîné en mode inductif (voir la Figure 3.24), ce qui le rend plus générique aux nouveaux documents non vus. Cela signifie que le $Multi - GAT_{DSS}$ n'a pas de visibilité sur les données du test pendant l'entraînement.

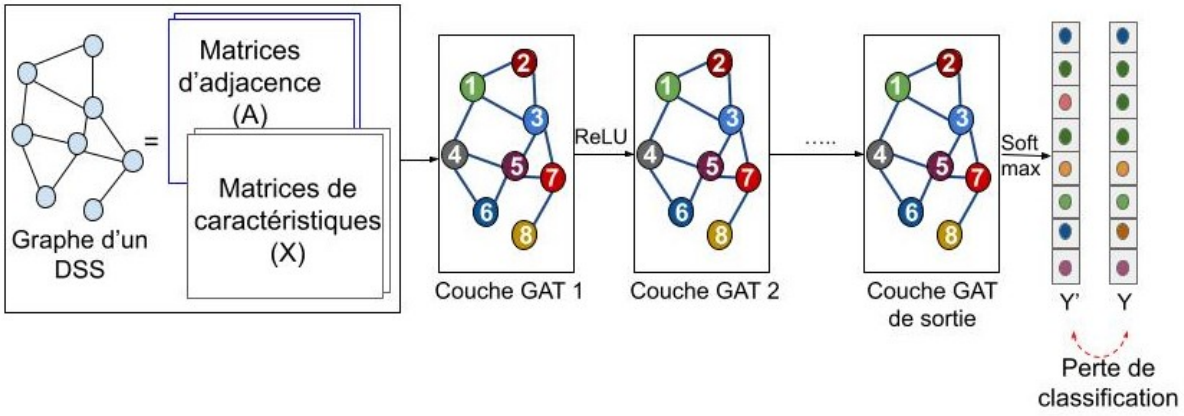


FIGURE 3.24 – Architecture Multi-GAT Inductive $Multi - GAT_{DSS}$.

3.4.5 Expérimentations et résultats

Dans l'implémentation du $Multi - GAT_{DSS}$, le ResNet-50 [53] Unet est adopté comme encodeur visuel pour construire le plongement de l'image. Toutes les expérimentations sont réalisées en utilisant une taille de mini-batch de 4. Le taux d'apprentissage est fixé à 0,002, le nombre maximum d'époques à 1000 et l'arrêt anticipé à 50.

Jeux de données utilisés

Dans cette partie, nous utilisons trois jeux de données des trois types de DSSs étudiés. Nous expérimentons le modèle $Multi - GAT_{DSS}$ sur le jeu de données réelles SROIE après correction des erreurs d'OCR et des erreurs d'annotation. Nous utilisons également un jeu de données Gen-Payslips de taille 1200 et nous générons artificiellement Gen-Invoices, un jeu de données de factures en français et en anglais, de taille 1200, contenant une partie des DSSs clonés et une autre, générée aléatoirement par notre générateur. Gen-Payslips et Gen-Invoices sont divisés en 700 documents pour l'entraînement, 200 pour la validation et 300 pour le test. La mise en page des trois jeux de données est variable à différents niveaux. Nous avons veillé à ce que le contenu et la mise en page soient aussi diversifiés que possible, au cours du processus de génération, afin que les données soient aussi réalistes que possible.

Évaluation des composantes du vecteur de caractéristiques

Pour montrer l'effet de chaque caractéristique dans le vecteur de caractéristiques, nous comparons les résultats du $Multi - GAT_{DSS}$ sur les trois jeux de données, en faisant varier les

composantes du vecteur de caractéristiques. Le vecteur de base contient initialement les caractéristiques textuelles, à savoir le Emb_{BPE} et le vecteur booléen Nature (8). Nous testons l'effet de l'absence et la présence des autres composantes, comme la position normalisée (Pos. Norm.), l'encodage de la région (Encod. Region) ainsi que les caractéristiques visuelles de l'image (Plong. Image) sur les prédictions du $Multi - GAT_{DSS}$. La Table 3.9 montre le score F1 obtenu sur SROIE, Gen-Invoices et Gen-Payslips, en faisant ces variations.

TABLE 3.9 – Score F1 (%) obtenu en faisant varier les caractéristiques du vecteur dans les trois jeux de données. Ces expériences sont réalisées à l'aide d'un $Multi - GAT_{DSS}$ à 4 couches. Dans la modélisation du graphe, n est fixé à 4 pour Gen-Payslips et Gen-Invoices et à 8 pour SROIE. « + » indique la présence et « - » l'absence de la caractéristique correspondante.

Caractéristiques ajoutées au vecteur de base			Jeux de données		
Pos. Norm.	Encod. Region	Plong. Image	SROIE	Gen-Invoices	Gen-Payslips
-	+	+	96.37	95.84	99.46
+	-	+	95.98	95.18	99.36
+	+	-	96.53	96.12	99.44
+	+	+	**97.77**	**96.87**	**99.48**

Comme le montre la Table 3.9, chaque caractéristique améliore le score de classification sur les trois jeux de données. La meilleure combinaison pour les trois jeux de données est le vecteur composé de la concaténation de : la nature du mot (Nature) ; du veteur de plongement textuel (BPEm ; la position normalisée (Pos. Norm.) ; de l'encodage de la région (Encod. Region) et du plongement de l'image (Plong. Image). Ces résultats confirment que la représentation multimodale permet d'améliorer les résultats de la classification des mots dans les DSS.

Évaluation du voisinage du mot dans le graphe

Nous avons également testé l'effet du voisinage du mot immédiat (du premier niveau) dans les graphes des trois jeux de données. Nous utilisons un $Multi - GAT_{DSS}$ à 4 couches pour les trois jeux de données. L'objectif de cette expérimentation est de trouver le nombre de voisins du premier niveau requis pour capturer le voisinage local suffisant pour la bonne reconnaissance des classes des mots dans le graphe.

TABLE 3.10 – Score F1 (%) obtenu avec le $Multi - GAT_{DSS}$ supervisé en variant le nombre de voisins du mot n dans le graphe.

Nombre de voisins (n)	SROIE	Gen-Invoices	Gen-Payslips
4	97.57	**96.87**	**99.48**
8	**97.77**	94.12	98.10
12	96.33	92.08	97.13

Les résultats de la Table 3.10 montrent que le modèle est plus performant avec n=4 (k=1) pour l'ensemble des données des factures et des fiches de paie, et autour de n=8 (k=2) pour le jeu de données SROIE. Cela peut s'expliquer par le fait que le premier niveau de voisinage des mots, qui est le plus important dans le DSS, doit contenir les mots-clés qui introduisent l'information. L'analyse du nombre de mots-clés introduisant les entités dans les trois jeux de

données révèle que, pour les jeux de données des fiches de paie et des factures, la plupart des mots-clés les plus forts aidant à la classification des noeuds sont des voisins du premier niveau des mots, notamment « ID », « Date », « No », « Number », « Total », etc.

La configuration de 8 voisins (deux à gauche, deux à droite, deux au-dessus et deux au-dessous du mot) donne le meilleur score pour SROIE. L'entité « Date » dans ce jeu de données, est généralement introduite par deux mots-clés : « Date : », étant donnée que la ponctuation « : » est considérée comme un mot séparé, le mot le plus indicatif est généralement le deuxième mot le plus proche de la date « Date ». La date peut être présentée sans mots-clés, mais les mots qui suivent la date sont des mots indicatifs également. Ils représentent habituellement l'heure, généralement composée de deux mots (« Heure 00 :00 :00 » ou « 00 :00 :00 AM »). L'entité « Total » est également souvent introduite par le mot-clé « Total » qui peut être le premier mot le plus proche à droite ou au-dessus du mot, mais dans plusieurs exemples, il s'agit du deuxième mot le plus proche, comme dans les expressions : « Total AMT », « Total (RM) », « Total Amount » et « Net Amt ».

Nous étudions également l'effet de la variation du nombre de couches dans le $Multi{-}GAT_{DSS}$ sur la prédiction des classes des nœuds du graphe, dans chaque jeu de données.

Comme le montre la Table 3.11, le modèle obtient les meilleurs résultats avec 4 couches pour les fiches de paie et les factures, et 6 pour SROIE. Ici, chaque couche prend en compte un niveau de voisinage plus élevé que celui de la précédente. Avec chaque nouvelle couche, nous prenons en considération des voisins plus éloignés, situés sur la même ligne et sur des lignes plus éloignées. À chaque nouvelle couche, nous ajoutons des informations provenant d'une autre ligne éloignée. Nous avons besoin d'au moins quatre niveaux de voisinage pour obtenir les meilleurs résultats. En effet, les informations représentées dans un DSS sont rangées dans des blocs d'informations, comme les informations sur l'entreprise (nom, adresse, identifiants, etc.), les identifiants de la commande (facture ou ticket de caisse), les informations sur le paiement qui suivent le total ou le détail du total qui le précède (TVA, remise, etc.). Ils forment ainsi un contexte local important pour la prédiction des entités.

TABLE 3.11 – Score F1 (%) obtenu en variant le nombre de couches dans le Multi-GAT.

Nb de couches	SROIE	Gen-Invoices	Gen-Payslips
2	95.99	91.54	98.06
3	97.56	93.88	99.25
4	97.77	**96.87**	**99.48**
5	97.85	96.62	99.20
6	**97.91**	96.59	99.05
7	97.45	95.70	98.72

En revanche, prendre un voisinage très grand ($\geq$ 5 pour Gen-Invoices et Gen-Payslips et $\geq$ 7 pour SROIE) peut provoquer une divergence. Cela conduit à la prise en considération d'un voisinage très lointain qui n'apporte aucune contribution à la classification des entités et peut même être perturbant pour la classification.

Évaluation de la matrice d'adjacence proposée

Nous comparons également dans cette section, les ressources et le temps de traitement issus de notre modélisation de la matrice d'adjacence ($Adj_{max\_nodes=256}$) avec deux autres modélisations : $Adj_{Dataset}$ et Adj_{DSS}. $Adj_{Dataset}$ modélise les relations de voisinage dans tous les documents de

la base de données. Adj_{DSS} modélise chaque document par une seule matrice d'adjacence en prenant le nombre maximum de noeuds dans la matrice, limité à 500, ce qui représente le nombre de noeuds maximum dans les documents du jeu de données. $Adj_{max\_nodes=256}$ correspond à la proposition que nous avons faite qui limite la taille de la matrice d'adjacence à 256 noeuds, avec la possibilité de modélisation d'un document par une ou plusieurs matrices d'adjacence.

TABLE 3.12 – Performances du $Multi - GAT_{DSS}$ à quatre couches GAT en variant la modélisation de la matrice d'adjacence sur SROIE.

	$Adj_{Dataset}$	Adj_{DSS}	$Adj_{max\_nodes=256}$
Dimensions de A	(60433,60433)	(500,500)	(256,256)
Temps de traitement d'un document (ms)	647	418	384

Sur une partie du jeu de données SROIE dont le total des mots est N= 60433, on calcule le temps de prédiction du $Multi - GAT_{DSS}$ (composé de quatre couches GAT) pour un seul document contenant 98 mots. Comme on peut le voir dans la Table 3.12, les deux réductions permettent de réduire le temps de traitement d'un seul document.

Dans $Adj_{Dataset}$, la taille de la matrice globale est proportionnelle à la taille du jeu de données d'apprentissage et de test. La matrice d'adjacence devient très volumineuse en apprenant sur un jeu de données plus grand, ce qui augmente l'espace mémoire nécessaire. Pour Adj_{DSS} qui est moins complexe et ne dépend pas du jeu de données d'apprentissage, le nombre maximal de noeuds dans les DSSs est différent d'un document à l'autre. Lors du traitement d'un DSS dont le nombre de mots dépasse n_max, cette approche risque de générer une erreur ou d'en éliminer et de ne pas traiter une partie des mots, ce qui entraîne une perte d'informations. Elle peut également nécessiter des modèles avec des matrices de dimensions différentes, dépendant du type de DSS. En utilisant $Adj_{max\_nodes=256}$, tous les documents peuvent être modélisés, peu importe le nombre de mots qu'ils contiennent. Le modèle d'apprentissage devient aussi plus léger et peut être unifié pour tous les types de documents.

Résultats globaux

Nous présentons les résultats finaux du modèle sur les différents jeux de données et nous les comparons également aux différents systèmes et méthodes de l'état de l'art, basés sur les modèles de convolution à graphe.

Comparaison des résultats avec les modèles de convolution à graphe sur Gen-Invoices
Nous comparons dans la Table 3.13, les performances des trois modèles basés sur les graphes, à savoir le GCN avec les 4 voisins [94], la première version transductive du $Multi - GAT_{Dataset}$ et la deuxième version inductive du $Multi - GAT_{DSS}$ avec le voisinage sur trois lignes. Nous soulignons que, comme le calcul des distances dh et dv, nécessaires pour la construction du voisinage étoile, génère des valeurs dispersées sur le nouveau jeu de données Gen-Invoices, nous adoptons le voisinage sur trois lignes pour $Multi - GAT_{Dataset}$.

Comme on peut le voir dans la Table 3.13, le modèle $Multi - GAT_{DSS}$ donne les meilleurs résultats. Il surpasse les deux modèles transductifs $Multi - GAT_{Dataset}$ et GCN [94]. Cela prouve que notre choix de calcul de caractéristiques multimodales, la modélisation des différentes structures du graphe ainsi que le choix de l'apprentissage inductif sur le $Multi - GAT_{DSS}$, améliorent la tâche d'extraction d'informations.

TABLE 3.13 – Comparaison du score F1 obtenu sur Gen-Invoices entre les différents modèles à graphes.

Système	GCN [94]	$Multi - GAT_{Dataset}$	$Multi - GAT_{DSS}$
Score F1 (%)	92.87	93.53	96,87

Résultats obtenus sur Gen-Payslips Nous évaluons les résultats détaillés du $Multi - GAT_{DSS}$ sur la classification des mots dans le jeu de données Gen-Payslips.

TABLE 3.14 – Précision (%), Rappel (%) et Score F1 (%) obtenus avec $Multi - GAT_{DSS}$ sur Gen-Payslips.

Classe	Précision	Rappel	Score F1
Employee name	97.06	93.36	95.18
Employment	94.94	97.40	96.16
Employee address	98.02	99.05	98.53
Employee code	94.44	100	97.14
Gross net pay	98.00	98.00	98.00
Social security number	99.19	96.85	98.01
Employee classification	99.14	97.88	98.51
Coefficient	98.88	98.88	98.88
Employee qualification	97.73	98.17	97.95
Starting date	97.67	98.99	98.33
Seniority	91.45	98.17	94.69
Payment date	100	99.17	99.58
Payment period	98.70	100	99.35
Company name	98.24	95.01	96.60
Earned leave	97.02	97.67	97.34
Company VAT number	97.75	98.81	98.28
Company address	97.92	99.11	98.51
Company APE code	95.77	98.00	96.87
Company SIRET number	98.47	98.60	98.53
Net salary	100	99.33	99.67
Company SIREN number	95.43	97.54	96.48
Net salary before taxes	100	100	100
Gross salary	99.33	99.33	99.33
Employee registration number	98.41	99.20	98.80
National collective agreement	98.86	98.19	98.52

Comme le montre la Table 3.14, tous les scores (Précision, Rappel et Score F1) obtenus sur toutes les entités sont supérieurs à 91%. Nous constatons également, que les entités « Employee registration number », « Employee qualification », « Earned leave », « coefficient » et « Company SIRET number » qui enregistrent des scores faibles avec le $Multi-GAT_{Dataset}$, obtiennent toutes des scores supérieurs à 95%. Cela prouve l'efficacité du modèle $Multi - GAT_{DSS}$ proposé.

Comparaison des résultats avec les systèmes de l'état de l'art sur SROIE Nous avons également comparé nos résultats sur SROIE aux résultats de plusieurs systèmes de l'état de l'art

dans la Table 3.15.

TABLE 3.15 – Comparaison du score F1 (%) entre le modèle $Multi - GAT_{DSS}$ et les systèmes de la littérature (M signifie million).

Système	# Paramètres	Score F1 (%)
LAMBERT[41]	125M	**98.17**
LayoutLMv2(L)[134]	426M	97.81
LayoutLMv2(B)[134]	200M	96.25
StrucText[84]	107M	96.88
ViBERTGrid[88]	157M	96.40
TRIE[145]	–	96.18
PICK[139]	–	96.12
LayoutLM(L)[135]	343M	95.24
LayoutLM(B)[135]	113M	94.38
BERT(B)[36]	340M	92
$Multi - GAT_{DSS}$	41M	**97.91**

$Multi - GAT_{DSS}$ obtient un score F1 important comparé aux différents systèmes de l'état de l'art. À l'inverse des autres modèles basés principalement sur l'architecture de Transformer et qui utilisent des centaines de millions de paramètres, $Multi - GAT_{DSS}$ contient uniquement 41 millions de paramètres (environ 40,6 millions pour les modèles pré-entraînés ResNet-UNet et BPEmb et 277 000 pour le Multi-GAT). Ce qui le rend plus rapide que les autres et beaucoup moins complexe que les autres systèmes.

3.4.6 Synthèse

Nous avons montré dans cette section que l'enrichissement des caractéristiques multimodales des mots du DSS dans le $Multi - GAT_{DSS}$ est très efficace. Contrairement aux approches transductives qui disposent d'une visibilité des données de test au cours de l'apprentissage, l'approche inductive proposée dans cette partie, permet de créer un modèle plus générique pour de nouveaux documents non vus au cours de la phase d'apprentissage. La nouvelle modélisation des matrices représentant les graphes des DSSs, en limitant leur dimension à une taille fixe, s'est également avérée moins coûteuse en termes de ressources (stockage mémoire et temps de traitement) et contribue beaucoup également à l'augmentation des performances du modèle. Nous avons en outre démontré que le voisinage immédiat du mot dans le DSS et le voisinage lointain dans le même bloc d'information contribuent tous les deux à la classification des nœuds. Le nombre de voisins immédiats requis est interprété par le nombre de mots-clés nécessaires pour reconnaître la classe de l'information. Toutefois, seuls les niveaux de voisinage qui font partie du même bloc d'information sont pris en considération dans le nombre de niveaux de voisinage requis. Ce voisinage éloigné permet de prendre en compte le contexte local indispensable dans le bloc d'information auquel appartient chaque mot, ce qui favorise également une meilleure classification.

Ce modèle, en revanche, est purement supervisé et n'exploite pas les documents non annotés pour améliorer ses performances. Nous souhaitons l'étendre au mode semi-supervisé en exploitant les documents non annotés appartenant au même domaine et en veillant à ne pas augmenter sa complexité.

3.5 Modèle d'EI semi-supervisé

Nous proposons ici une extension du modèle $Multi - GAT_{DSS}$ en mode semi-supervisé. Nous ajoutons au Multi-GAT supervisé un module génératif permettant au système d'apprendre simultanément à partir de données annotées et non annotées, renforçant ainsi les performances du $Multi - GAT_{DSS}$. Ce module génératif est l'auto-encodeur variationnel à graphe (VGAE). Comme le montre la Figure 3.25, cette approche semi-supervisée se compose principalement de trois parties : la modélisation par graphe, le classifieur Multi-GAT et le VGAE.

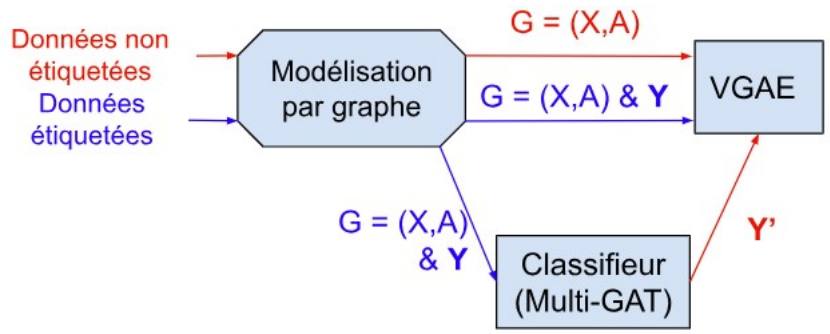


FIGURE 3.25 – Architecture du modèle d'EI semi-supervisé avec le flux de données étiquetées et non étiquetées à travers les trois composants de l'architecture (modélisation par graphe, classifieur et VGAE).

La modélisation par graphe et le classifieur correspondent au modèle utilisé dans l'architecture précédente ($Multi - GAT_{DSS}$).

Nous décrivons dans cette section le composant VGAE, l'architecture semi-supervisée globale ainsi que la fonction objective du système global Multi-GAT+VGAE.

3.5.1 Auto-encodeur variationnel à graphe

L'Auto-encodeur variationnel à graphe (Variational Graph Auto-Encoder (VGAE)) est composé d'un encodeur et de deux décodeurs, comme le montre la Figure 3.26. L'encodeur est principalement constitué de couches GAT et de couches denses. Il partage ses premières couches GAT avec le classifieur et produit des variables latentes gaussiennes Z. Ces couches partagées permettent d'associer la tâche de classification à l'apprentissage de la distribution des données. Le classifieur devient ainsi plus générique au fur et à mesure qu'il apprend les caractéristiques des données dans leur distribution. L'encodeur apprend d'abord le mappage des données d'entrée dans l'espace latent. La distribution des variables latentes est modélisée par une distribution gaussienne dont la moyenne et la variance sont générées par les couches de sortie de l'encodeur. À partir de ces variables gaussiennes et du résultat du classifieur (Y') ou de la vérité terrain (Y), nous reconstruisons les matrices X et A à l'aide de deux décodeurs. Le premier décodeur qui reconstruit A est constitué principalement d'une couche de produit scalaire, comme proposé dans [77], tandis que le second décodeur qui reconstruit X, est constitué de plusieurs couches de convolution 2D.

L'entrée des décodeurs est la concaténation de la variable latente Z générée par l'encodeur et du vecteur des classes de nœuds, comme le montre la Figure 3.26. Le vecteur des classes de nœuds est soit Y qui représente la « vérité terrain (GT : Ground Truth) » si l'entrée est étiquetée, soit Y' qui représente les prédictions des classes de nœuds si l'entrée n'est pas étiquetée. Les décodeurs tentent de reconstruire l'entrée à partir de la variable latente Z et du vecteur (Y/Y').

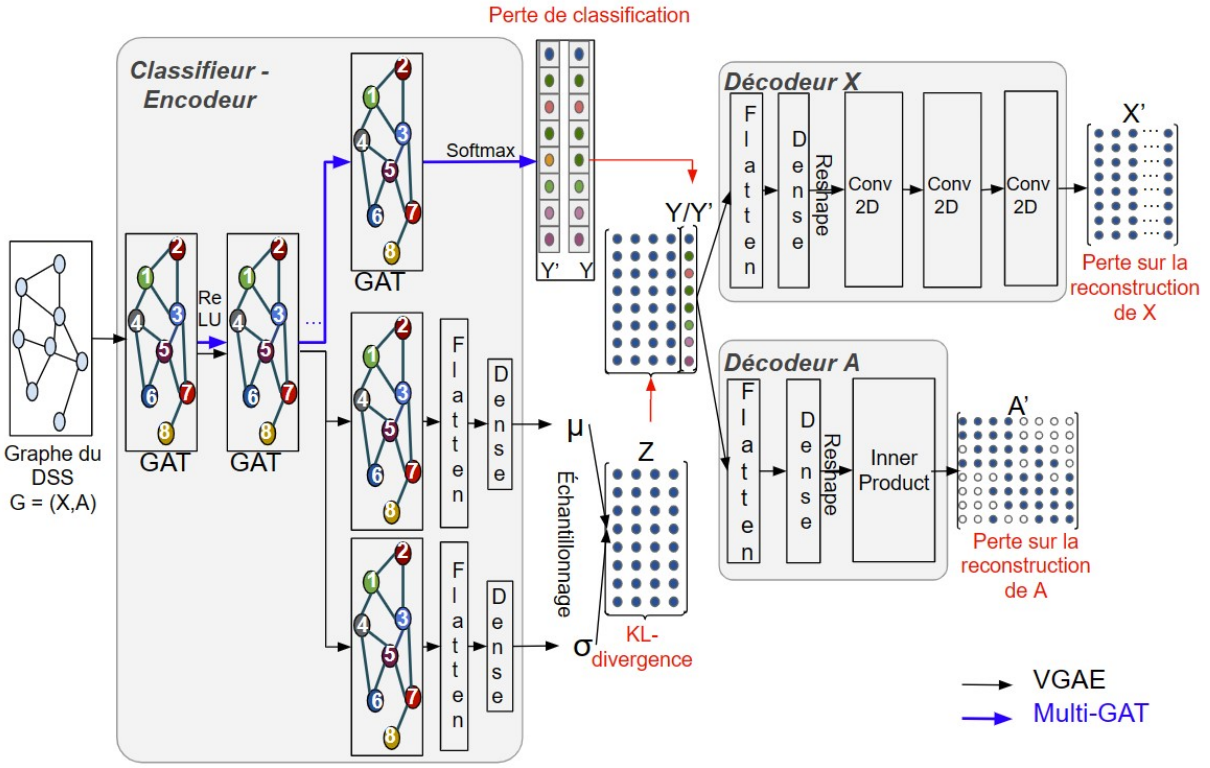


FIGURE 3.26 – Vue d'ensemble du système semi-supervisé Multi-GAT+VGAE. Le graphe DSS est introduit dans le classifieur et l'encodeur du VGAE. Ces deux modules partagent leurs premières couches GAT. Le classifieur fournit en sortie le vecteur de prédiction des nœuds du graphe (Y') et l'encodeur produit à son tour la variable latente Z. Ces deux sorties sont ensuite concaténées et introduites dans les deux décodeurs du VGAE : Décodeur X et Décodeur A qui cherchent à reconstruire les matrices X et A. Le décodeur X est composé principalement de couches de Conv 2D alors que le décodeur A est composé essentiellement d'une couche de produit scalaire. Les quatre termes de la fonction de perte calculés sur la sortie de chaque composant, sont également représentés ici en rouge.

Les prédictions du classifieur affectent directement la sortie des deux décodeurs. Lorsque les graphes d'entrée sont des données non étiquetées, les poids du classifieur sont mis à jour en utilisant les pertes VGAE. Cela signifie que notre système apprend des caractéristiques utiles à partir de données étiquetées et non étiquetées.

Pour ne pas biaiser les résultats du VGAE et s'assurer que l'entrée du décodeur soit aussi précise que possible, nous commençons d'abord par entraîner le classifieur sur les DSSs étiquetés, puis nous poursuivons l'entraînement en ajoutant les autres composants du VGAE ainsi que les DSSs non étiquetés.

3.5.2 Le problème d'inférence

L'architecture proposée traite un problème d'optimisation. Notre but est de trouver les paramètres du modèle qui minimisent une fonction objective (la fonction de perte). Ce problème peut être exprimé comme un problème d'inférence : déduire la valeur d'une variable aléatoire à partir de la valeur d'une autre variable (formule de Bayes).

Dans le problème d'inférence associé à cette architecture, tous les composants, à savoir l'encodeur, les décodeurs et le classifieur sont considérés comme des distributions de probabilités. Les probabilités qui décrivent les distributions des variables données par chaque composant sont les suivantes :

— l'encodeur : $p(Z/X, A)$

— le décodeur qui reconstruit X : $p(X/Z, Y)$

— le décodeur qui reconstruit A : $p(A/Z, Y)$

— le classifieur : $p(Y/X, A)$

Nous utilisons le théorème de Bayes qui relie la probabilité a priori $p(Z)$ à la vraisemblance $p(X/Z, Y)$ et $p(A/Z, Y)$ et à la probabilité a posteriori $p(Z/X, A)$. $p(Z)$ correspond à la distribution gaussienne. A et X (les données d'entrée) sont distribuées selon une loi de probabilité inconnue. Le modèle que nous proposons est représentatif de la base de données (X, A) si : pour chaque $x \in X$ et $\forall a \in A$, il existe au moins une configuration de la variable latente Z qui permet au modèle de générer x' et a' (similaires ou très proches de x et a).

Pour s'assurer que le modèle représente la base de données, nous cherchons à maximiser la log-vraisemblance des données : $\log p(X, A)$ qui peut être exprimée comme une fonction d'autres termes $p(X, A) = \frac{p(X,A/Z).p(Z)}{p(Z|A,X)}$ (Formule de Bayes).

3.5.3 Optimisation

Notre système propose une fonction de perte paramétrée en fonction du type de données d'entrée et des pertes associées à chaque composant élémentaire (classifieur, encodeur, décodeur). La fonction de perte globale du système contient quatre termes : la perte de classification, la perte de reconstruction X, la perte de reconstruction A et enfin la perte de la KL-Divergence (Divergence Kullabck-Leibler). Cette dernière mesure la différence entre les variables latentes Z et la distribution gaussienne. L'objectif est de minimiser la perte de KL-Divergence pour adapter la distribution des variables latentes à la distribution gaussienne. Cette perte globale est utilisée pour optimiser les paramètres de toutes les parties du modèle (classifieur, encodeur et les deux décodeurs). Pour les données non étiquetées, nous ne calculons que les pertes de reconstruction et la perte de KL-Divergence, avec lesquelles nous optimisons tous les composants (même le classifieur). En s'appuyant sur la limite de vraisemblance marginale proposée par [75], nous détaillons dans la suite de cette section, le calcul de la fonction objective en utilisant l'équivalence d'inférence.

Pour les données étiquetées, considérons $x \in X$, $a \in A$ et $y \in Y$ comme des variables aléatoires. L'objectif de notre modèle est de maximiser la log-vraisemblance $\log p_\theta(x, a, y)$. Nous cherchons à trouver le paramètre optimal θ^* qui maximise $p(x, a, y)$. θ^* maximise la probabilité de générer des échantillons de données corrects $\theta^* = \arg\max \prod_i^n p_\theta(x, a, y)^i$. Nous utilisons ensuite le logarithme pour convertir le produit en somme : $\theta^* = \arg\max \sum_i^n \log p_\theta(x, a, y)^i$.

Comme il est difficile de maximiser directement $\log p_\theta(x, a, y)$, nous maximisons sa borne inférieure variationnelle $L(x, a, y)$ (ELBO) ($\log p_\theta(x, a, y) \geq L(x, a, y)$)

Sachant que « p » et « q » sont deux distributions et que par définition, nous avons : $\log p(x) =$

$\log \int_z p(x,z)$, alors nous avons le développement suivant :

$$\begin{aligned}
\log p(x,a,y) &= \log \int_z p(x,a,y,z) \\
&= \log \int_z p(x,a,y,z) \frac{q(z \mid x,a,y)}{q(z \mid x,a,y)} \\
&= \log \int_z q(z \mid x,a,y) \frac{p(x,a,y,z)}{q(z \mid x,a,y)} \\
&= \log E_{q(z|x,a,y)} \frac{p(x,a,y,z)}{q(z \mid x,a,y)}
\end{aligned} \tag{3.10}$$

En appliquant les formules : $\log E(*) \geq E(\log *)$ (première ligne) ; $p(a,b,c) = p(a \mid b,c)p(b \mid c)p(c)$ (deuxième ligne) et $log(a.b) = log(a) + log(b)$ et $log(\frac{a}{b}) = log(a) - log(b)$ (dernière ligne), nous obtenons :

$$\begin{aligned}
\log p(x,a,y) \geq & E_{q(z|x,a,y)} \left[\log \frac{p(x,a,y,z)}{q(z \mid x,a,y)} \right] \\
\geq & E_{q(z|x,a,y)} \left[\log \frac{p(x \mid a,y,z)p(a \mid y,z).p(y \mid z)p(z)}{q(z \mid x,a,y)} \right] \\
\geq & E_{q(z|x,a,y)} \left[\log p(x \mid y,z) \right] + E_{q(z|x,a,y)} \left[\log p(a \mid y,z) \right] + E_{q(z|x,a,y)} \left[\log p(y) \right] \\
& + E_{q(z|x,a,y)} \left[\log \frac{p(z)}{q(z \mid x,a,y)} \right] = L(x,a,y)
\end{aligned} \tag{3.11}$$

Ainsi, pour les données étiquetées, nous avons :

$$\begin{aligned}
\log p_\theta(x,a,y) \geq & E_{q_\phi(z|x,a,y)} \left[\log p_\theta(x \mid y,z) \right] + E_{q_\phi(z|x,a,y)} \left[\log p_\theta(a \mid y,z) \right] + E_{q_\phi(z|x,a,y)} \left[\log p_\theta(y)) \right] \\
& + E_{q_\phi(z|x,a,y)} \left[\log \frac{p_\theta(z)}{q_\phi(z \mid x,a,y)} \right]
\end{aligned}$$

Nous remplaçons le dernier terme par son équivalent :

$$\begin{aligned}
\log p_\theta(x,a,y) \geq & E_{q_\phi(z|x,a,y)} \left[\log p_\theta(x \mid y,z) \right] + E_{q_\phi(z|x,a,y)} \left[\log p_\theta(a \mid y,z) \right] + \log p_\theta(y) \\
& + DKL(q_\phi(z \mid x,a,y) \mid\mid p_\theta(z))
\end{aligned} \tag{3.12}$$

DKL correspond à la KL-Divergence de $p_\theta(z)$ et $q_\phi(z \mid x,a,y)$. $E_{q_\phi(z|x,a,y)} \left[\log p_\theta(x \mid y,z) \right]$ et $E_{q_\phi(z|x,a,y)} \left[\log p_\theta(a \mid y,z) \right]$ correspondent à l'erreur sur la reconstruction de x et de a respectivement. On essaie de rendre le x et le a générés plus corrélés à la variable latente z.

Nous introduisons un paramètre β afin d'orienter davantage la perte sur la variable latente gaussienne.

$$\begin{aligned}
\log p_\theta(x,a,y) \geq & E_{q_\phi(z|x,a,y)} \left[\log p_\theta(x \mid y,z) \right] + E_{q_\phi(z|x,a,y)} \left[\log p_\theta(a \mid y,z) \right] + \log p_\theta(y) \\
& + \beta DKL(q_\phi(z \mid x,a,y) \mid\mid p_\theta(z))
\end{aligned} \tag{3.13}$$

Pour les données non étiquetées, l'objectif est de maximiser l'ELBO de la probabilité margi-

nale $p_\theta(x, a)$ et le label y est considéré comme une variable discrète latente.

$$\begin{aligned}
\log p_\theta(x,a) \geq & E_{q_\phi(y,z|x,a)}[\log p_\theta(x \mid y,z) + \log p_\theta(a \mid y,z) + \log p_\theta(y) + \log p_\theta(z) - \log q_\phi(y,z \mid x,a)] \\
\geq & E_{q_\phi(z|y,x,a)q_\phi(y|x,a)}[\log p_\theta(x \mid y,z) + \log p_\theta(a \mid y,z) + \log p_\theta(z) + \log p_\theta(y) \\
& - \log(q_\phi(z \mid y,x,a)q_\phi(y \mid x,a))] \\
\geq & \sum_y q_\phi(y \mid x,a)(E_{q_\phi(z|y,x,a)}[\log p_\theta(x \mid y,z) + \log p_\theta(a \mid y,z) + \log p_\theta(y) + \log p_\theta(z) \\
& - \log q_\phi(z \mid y,x,a) - \log q_\phi(y \mid x,a)]) \\
\geq & \sum_y q_\phi(y \mid x,a)(L(x,a,y) - E_{q_\phi(z|y,x,a)}\left[\log q_\phi(y \mid x,a)\right]) \\
= & \sum_y q_\phi(y \mid x,a)\left(L(x,a,y) - \log q_\phi(y \mid x,a)\right) \\
= & \sum_y q_\phi(y \mid x,a)\left(L(x,a,y)\right) - H\left[q_\phi(y \mid x,a)\right] \\
= & U(x,a)
\end{aligned} \tag{3.14}$$

où :

$$H\left[q_\phi(y \mid x,a)\right] = \sum_y q_\phi(y \mid x,a)\left[\log q_\phi(y \mid x,a)\right]$$

Lorsque nous calculons la fonction de perte globale, nous ajoutons à la borne inférieure variationnelle $L(x, a, y)$, la perte de classification sur les documents étiquetés définie comme suit : $E_{(x,a,y)\sim DL}\left[\log q_\phi(y \mid x,a)\right]$

Maximiser les fonctions $L(x, a, y)$ et $U(x, a)$ revient à minimiser (-L) et (-U). La fonction objectif globale à minimiser est :

$$J = E_{(x,a,y)\sim DL}\left[-L(x,a,y)\right] + E_{(x,a)\sim DU}\left[-U(x,a)\right] + \alpha E_{(x,a,y)\sim DL}\left[\log(q_\phi(y \mid x,a))\right] \tag{3.15}$$

où DL est la donnée étiquetée, DU est la donnée non étiquetée et α et β sont des paramètres qui équilibrent le poids relatif entre l'entraînement du modèle génératif et du modèle discriminatif (classifieur).

Nous proposons un paramétrage de β basé sur la perte de classification $\beta = f(E_{(x,a,y)\sim LD}\left[\log(q_\phi(y \mid x,a))\right])$, et $f(x) = 1 + \frac{l}{u} * x$. l se réfère à la taille de LD et u à la taille de UD. L'objectif principal de ce paramètre est de pénaliser la perte KL en fonction de la perte de classification et de forcer le Multi-GAT+VGAE à apprendre une meilleure estimation de la distribution de z, ce qui améliore la tâche de classification. En outre, nous modifions le α introduit dans [75] en le fixant à $\frac{l}{u}$, afin de donner la priorité à la perte de classification.

En proposant un encodeur et un classifieur qui partagent leurs premières couches GAT et en paramétrant la perte de VGAE en fonction de la perte de classification, le VGAE s'oriente le plus possible vers l'optimisation de la tâche de classification.

3.5.4 Expérimentations et résultats

Le taux d'apprentissage est fixé à 0,002 pour le Multi-GAT et à $1e^{-5}$ pour le VGAE. La dimension de la variable latente utilisée comme sortie de l'encodeur est fixée à 5. Le nombre maximum d'époques est gardé à 1000 et l'arrêt précoce (early stopping) est fixé à 50 étapes.

Les jeux de données utilisés

Nous expérimentons le modèle semi-supervisé Multi-GAT+VGAE sur le jeu de données réel SROIE ainsi que sur les deux autres jeux de données générés : Gen-Invoices et Gen-Payslips utilisés pour tester le $Multi-GAT_{DSS}$. Nous alimentons également chaque jeu de données avec 300 documents non annotés générés artificiellement. Nous appelons Gen-Receipts est l'ensemble des tickets de caisse générés. Les détails concernant les trois jeux de données sont illustrés dans la Table 3.16.

TABLE 3.16 – Statistiques et caractéristiques des jeux de données, * fait référence à des documents réels.

Caractéristique	SROIE	Gen-Invoices	Gen-Payslips
Taille de l'ensemble d'entraînement	626*	700	700
Taille de l'ensemble de validation	-	200	200
Taille de l'ensemble de test	437*	300	300
Taille de l'ensemble non étiqueté	300	300	300
Nombre d'entités	4	27	25
Langue	Anglais	Anglais & Français	Français

Modes d'apprentissage

Nous étudions l'effet du mode semi-supervisé sur les performances du modèle. Nous comparons les scores F1 obtenus sur les deux jeux de données générés (Gen-Invoices et Gen-Payslips) en utilisant le modèle supervisé $Multi-GAT_{DSS}$, deux modèles de base VGAE semi-supervisés et le modèle semi-supervisé final (Multi-GAT+VGAE). Multi-GAT+VGAE$_{B_1}$ est le Multi-GAT+VGAE sans les couches GAT partagées entre l'encodeur et le classifieur, et la paramétrisation α/β. Multi-GAT+VGAE$_{B_2}$ est le Multi-GA+VGAE sans l'hyperparamètre de régulation β.

TABLE 3.17 – Score F1 (%) obtenu en variant le mode d'apprentissage, le Multi-GAT contient 4 couches pour Gen-Invoices et Gen-Payslips (dans les hyperparamètres α et β, l se réfère à la taille de l'ensemble des données étiquetées, u à l'ensemble des données non étiquetées et L_s est la perte de classification).

Modèle	GATs partagés	α	β	Gen-Invoices	Gen-Payslips
$Multi-GAT_{DSS}$	-	-	-	96.87	99.48
Multi-GAT+VGAE$_{B_1}$	✗	1	1	97.14	99.56
Multi-GAT+VGAE$_{B_2}$	✓	l/u	1	97.27	99.56
Multi-GAT+VGAE	✓	l/u	$1+(l/u)*L_s$	**97.34**	**99.58**

Comme le montre la Table 3.17, le Multi-GAT+VGAE surpasse le $Multi-GAT_{DSS}$ supervisé et les deux modèles Multi-GAT+VGAE$_{B_1}$ et Multi-GAT+VGAE$_{B_2}$. Cela certifie que notre architecture avec des couches GAT partagées, est avantageuse et que notre fonction de perte proposée optimise mieux les résultats de la classification. En utilisant le jeu de données Gen-Receipts, nous améliorons également légèrement les résultats de SROIE avec un score F1 de 97,94 %. Compte tenu du fait que la présentation des tickets de caisse générés diffère légèrement de celle de SROIE, l'amélioration est moins importante que pour les deux autres jeux de données.

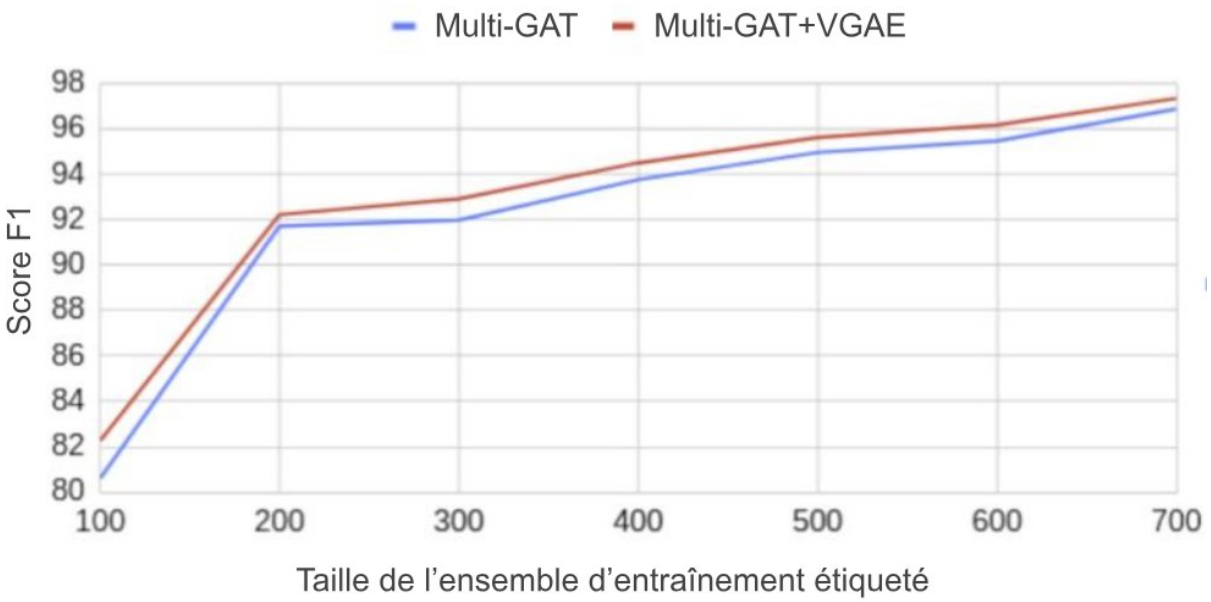


FIGURE 3.27 – Score F1 (%) obtenu avec Multi-GAT ($Multi-GAT_{DSS}$) et Multi-GAT+VGAE en variant la taille de l'ensemble d'entraînement des factures étiquetées.

Nous varions la taille du jeu de données d'entraînement étiqueté dans Gen-Invoices et nous comparons les résultats du Multi-GAT à ceux du Multi-GAT+VGAE. La Figure 3.27 montre que le modèle Multi-GAT+VGAE améliore le score F1 obtenu par le Multi-GAT supervisé ($Multi - GAT_{DSS}$) sur Gen-Invoices pour toutes les tailles des jeux de données étiquetées. Malgré le nombre limité de données étiquetées, le score F1 passe de 80,58% à 82,26% avec Multi-GAT+VGAE pour le plus petit jeu de données de 100 factures. Cela prouve que le modèle Multi-GAT+VGAE peut améliorer le score de prédiction même avec des données étiquetées limitées.

Résultats globaux

Nous avons comparé les résultats obtenus par Multi-GAT+VGAE sur les trois jeux de données, avec ceux des deux systèmes de prédiction de nœuds de graphe à savoir : le GCN [94] et $Multi - GAT_{Dataset}$ avec toutefois notre nouveau calcul du voisinage dans le graphe. Comme le montre la Table 3.18, notre système est plus performant que les deux autres. Il est plus riche en caractéristiques de mots et fournit un voisinage de nœuds de graphe plus significatif. En outre, il intègre des DSSs non étiquetés dans un schéma semi-supervisé.

TABLE 3.18 – Score F1 (%) obtenu avec les systèmes de prédiction des nœuds du graphe.

Modèle	SROIE	Gen-Invoices	Gen-Payslips
GCN [94]	96.71	92.87	97.98
$Multi - GAT_{Dataset}$	97.62	93.53	99.28
Multi-GAT+VGAE	**97.94**	**97.34**	**99.58**

Nous remarquons que les scores enregistrés pour Gen-Payslips sont les plus élevés, cela résulte

du fait que ce jeu de données a moins de variation dans le contenu et la mise en page que les deux autres.

Nous avons également comparé nos résultats (score F1) et la complexité du modèle (nombre de paramètres) sur SROIE avec plusieurs systèmes très performants de la littérature (voir Table 3.19).

TABLE 3.19 – Comparaison du score F1 (%) entre notre système et les systèmes de la littérature (M signifie million).

Système	# paramètres	Score F1 (%)
LAMBERT[41]	125M	**98.17**
LayoutLMv2(L)[134]	426M	97.81
LayoutLMv2(B)[134]	200M	96.25
StrucText[84]	107M	96.88
ViBERTGrid[88]	157M	96.40
TRIE[145]	–	96.18
PICK[139]	–	96.12
LayoutLM(L)[135]	343M	95.24
LayoutLM(B)[135]	113M	94.38
BERT(B)[36]	340M	92
Multi-GAT+VGAE	41M	**97.94**

Notre système obtient des résultats compétitifs ; il arrive en deuxième position après le modèle le plus performant [41]. Nous tenons à souligner que le meilleur modèle enregistré sur la Table de classement public pour l'extraction d'informations clés de SROIE appartient à StructText [84] avec 98,70 %. Notre système d'inférence reste moins complexe que les autres systèmes, le classifieur ne contenant qu'environ 41 millions de paramètres. Notre système est donc plus rapide que les autres. De plus, contrairement aux autres, notre architecture optimise les résultats de la classification en exploitant les données non étiquetées du même domaine sans augmenter le nombre de paramètres du modèle d'inférence final. Il est important de noter que les composants du VGAE ne sont utilisés que dans la phase d'apprentissage, tandis que seul le Multi-GAT est utilisé dans l'étape d'inférence.

3.5.5 Synthèse

Dans cette section, nous avons présenté une extension du classifieur de nœuds de graphes $Multi-GAT_{DSS}$ en mode semi-supervisé, basée sur une architecture qui combine le $Multi-GAT_{DSS}$ avec un auto-encodeur variationnel à graphe (VGAE). Nous avons intégré un VGAE qui optimise le classifieur avec des données étiquetées et non étiquetées dans le domaine. Le VGAE est orienté le plus possible vers l'optimisation de la tâche de classification en proposant un encodeur et un classifieur qui partagent leurs premières couches GAT et en paramétrant la perte VGAE en fonction de la perte de classification. Le système Multi-GAT+VGAE a surpassé les deux versions supervisées du Multi-GAT, le GCN et d'autres systèmes semi-VGAE de base sur les jeux de données Gen-Invoices et Gen-Payslips.

3.6 Conclusion

En raison du manque de jeux de données de type DSS pour l'entraînement des modèles d'IE, nous avons d'abord conçu une méthode de génération des DSSs. Nous avons ensuite généré des jeux de données de trois types de DSS : factures, fiches de paie et tickets de caisse. Ces jeux de données ont été évaluées à l'aide de plusieurs métriques d'évaluation de la diversité du contenu et de la mise en page.

Nous avons ensuite présenté des versions évolutives du système d'EI. Nous avons adopté un modèle basé sur la représentation du DSS comme un graphe de mots et un classifieur basé sur le mécanisme d'attention multi-têtes. Le premier modèle proposé est le $Multi - GAT_{Dataset}$, qui est un modèle supervisé transductif avec un calcul de voisinage par graphe étoile. Toutefois, ce modèle présente des limites en termes de complexité de calcul du voisinage dans certains cas et de généricité de la méthode d'apprentissage, qui dispose d'une visibilité sur les données de test. Le modèle $Multi - GAT_{DSS}$ a ensuite été présenté en proposant une nouvelle méthode de calcul de voisinage plus générique, basée sur un calcul de voisinage sur trois lignes, un mode d'apprentissage inductif et une représentation matricielle du graphe simplifiée. En vue d'améliorer les performances du modèle $Multi - GAT_{DSS}$, les données non annotées ont été exploitées en étendant le modèle au mode semi-supervisé par le biais d'une combinaison avec un VGAE, ce qui a donné lieu au modèle $Multi - GAT + VGAE$.

4

Optimisation et adaptation du Multi-GAT au chiffrement homomorphe

Sommaire

4.1 Introduction

L'objectif recherché dans ce chapitre est double : optimiser la représentation des données et adapter le modèle d'inférence à une sécurité optimale pour son exécution sur des serveurs distants. Pour le premier point, nous cherchons à optimiser la représentation des caractéristiques des mots dans le graphe tout en améliorant les performances du $Multi - GAT$. Pour cela, nous commençons par présenter un processus de réduction de la dimensionnalité des données et de la complexité du modèle. Il s'agit d'éliminer les caractéristiques redondantes et non informatives qui peuvent baisser les performances du modèle, tels que les paddings dans les vecteurs de plongements des sous-mots. Cela implique également de proposer une méthode efficace pour fusionner les différentes modalités de caractéristiques.

Pour le second point, comme il est prévu de fournir un modèle sécurisé dans le cadre du projet, il est nécessaire de penser à garantir la confidentialité du modèle et des données des

utilisateurs lors de l'analyse de leurs documents sur un serveur distant. A cet effet, nous allons montrer comment adapter le $Multi-GAT$ aux protocoles de chiffrement homomorphe qui vont permettre d'assurer cette confidentialité lors du passage sur un serveur distant. Ces protocoles nécessitent l'approximation polynomiale des opérations non homomorphes (non-linéaires) dans le $Multi-GAT$. Ils bénéficient également de la réduction de la dimensionnalité des données en vue d'optimiser le temps de leur traitement. Tous ces aspects seront discutés et illustrés dans ce chapitre.

4.2 Réduction de la dimensionnalité des données

4.2.1 Introduction

Lorsque la dimension des données augmente, les performances des classifieurs neuronaux s'améliorent, mais lorsque cette dimension continue d'augmenter, leurs performances se dégradent [68]. La réduction de la dimensionnalité des données est par conséquent un moyen nécessaire pour remédier à ces problèmes. Cela revient à ramener la représentation des données d'un espace à haute dimension à un espace à plus faible dimension.

Dans cette section, nous présenterons d'abord l'état de l'art sur les méthodes de réduction de la dimensionnalité, puis nous expliquerons notre approche proposée.

4.2.2 État de l'art sur la réduction de la dimensinnalité des données

Les jeux de données à haute dimension (avec un grand nombre de caractéristiques) contiennent souvent des informations redondantes ou non pertinentes. La réduction de la dimensionnalité vise à éliminer ces éléments indésirables [68]. Plusieurs travaux dans la littérature ont proposé différentes approches de réduction de la dimension des données. Ces approches correspondent à des processus d'extraction et de sélection de caractéristiques [141].

Sélection de caractéristiques

La sélection de caractéristiques est une méthode qui consiste à sélectionner des sous-ensembles de caractéristiques qui sont utilisés pour la constitution d'un modèle [30]. Les techniques correspondantes requièrent des données comportant de nombreuses caractéristiques redondantes ou similaires qui peuvent être supprimées sans perte significative d'informations. Elles sont fréquemment utilisées lorsque les caractéristiques sont nombreuses et les échantillons de données relativement limités [29, 30].

La sélection de caractéristiques peut être séparée en deux phases : la génération de sous-ensembles de caractéristiques et l'évaluation des sous-ensembles générés (s'ils sont optimaux ou non) [73, 127]. Elle peut être considérée comme un problème d'optimisation de la recherche [68]. Pour définir le sous-ensemble optimal de caractéristiques, il convient de rechercher les M combinaisons de caractéristiques parmi les N caractéristiques initiales possibles.

Langley et al. [80] a regroupé les différentes méthodes de sélection de caractéristiques en deux grandes catégories (à savoir le filtre et l'enveloppe) en fonction de leur dépendance à l'égard de l'algorithme de classification qui utilisera le sous-ensemble sélectionné. Les méthodes de filtre sont indépendantes de l'algorithme d'apprentissage. Elles évaluent les caractéristiques en se basant sur leurs propriétés intrinsèques avant les tâches d'apprentissage. Elles utilisent quatre types de critères de mesure différents, à savoir l'information, la dépendance, la cohérence et la distance

[30]. Le principal inconvénient de ces méthodes est qu'elles négligent le lien entre le sous-ensemble sélectionné et les performances de l'algorithme d'apprentissage [1, 69].

Les méthodes d'enveloppe, quant à elles, enveloppent la sélection de caractéristiques autour de l'algorithme d'apprentissage et utilisent le taux d'erreur du processus de classification comme critère d'évaluation des caractéristiques [141]. Elles sélectionnent les caractéristiques les plus optimales pour l'algorithme de prédiction. Elles permettent donc d'obtenir de meilleures performances et une grande précision comparée aux algorithmes de filtre [65, 69].

On retrouve aussi la méthode intégrée, qui est un processus de sélection de caractéristiques intégré dans l'algorithme d'apprentissage. Elle utilise les propriétés du classifieur pour orienter l'évaluation des caractéristiques [30, 141]. Cette méthode évite ainsi l'exécution répétée du classifieur pour l'évaluation de chaque sous-ensemble de caractéristiques. Elle est donc moins coûteuse en ressources de calcul comparé à la méthode de l'enveloppe.

D'autres travaux combinent les différentes méthodes comme Peng et al. [111] qui combinent la méthode de filtre et enveloppe.

Extraction de caractéristiques

L'extraction de caractéristiques permet de générer de nouvelles caractéristiques à partir des caractéristiques d'origine, ce qui se traduit par un mappage des caractéristiques d'origine vers les nouvelles caractéristiques. Cette approche permet de compresser plus efficacement les nouvelles caractéristiques. Cependant, les nouvelles caractéristiques peuvent perdre leur signification si l'ensemble des caractéristiques d'origine a une interprétation claire [55].

Parmi les nombreuses techniques de réduction de la dimensionnalité et d'extraction des caractéristiques linéaires, la plus intuitive est la projection, qui permet de projeter les données d'un espace de caractéristiques à haute dimension vers un espace de plus faible dimension [59]. L'analyse en composantes principales « ACP » (ou la PCA : Principal Component Analysis) [2] est la technique de projection la plus répandue pour la réduction de la dimensionnalité. L'ACP est une méthode d'apprentissage non supervisée qui réduit les caractéristiques en maximisant leur variance dans une dimension plus faible. L'ACP utilise une combinaison linéaire de variables pour représenter les composantes principales des données de départ. Les composantes principales doivent refléter autant que possible l'information contenue dans les données originales, et ces composantes principales doivent être indépendantes les unes des autres [2, 104]. Cet algorithme peut être résumé par les étapes suivantes : la normalisation des données brutes (les données doivent avoir une variance unitaire et une moyenne nulle) ; le calcul de la matrice de co-variance des données ; le calcul des vecteurs et valeurs propres de la matrice de covariance. Les données brutes sont alors projetées dans un sous-espace à k dimensions constitué des k premiers vecteurs propres de la matrice de co-variance, en constituant ainsi la nouvelle base des données.

D'autres travaux utilisent également LDA (Linear Discriminant Analysis) ou des combinaison ACP et LDA [109, 148] pour l'extraction de caractéristiques.

L'algorithme LDA est conçu pour regrouper de manière optimale différentes classes d'objets selon une projection dans un sous-espace de faible dimension [38]. La tâche principale de LDA est de convertir l'échantillon original par projection dans le meilleur espace vectoriel discriminant pour contribuer à l'extraction des informations de classification et à la réduction des dimensions, de sorte que les échantillons de données après projection aient la plus grande distance inter-classes et la plus petite distance intra-classe (matrice de dispersion inter-classes maximale et matrice de dispersion intra-classe minimale) [68]. Le calcul de la LDA comporte plusieurs étapes. Tout d'abord, un vecteur moyen est calculé pour chaque classe du jeu de données. Ensuite, les matrices de dispersion sont calculées et les vecteurs propres et valeurs propres correspondants de

ces matrices sont déterminés. Les vecteurs propres sont triés par ordre décroissant des valeurs propres et les k premiers vecteurs propres sont sélectionnés pour former une matrice. Enfin, cette matrice est utilisée pour projeter les échantillons d'entrée dans le nouveau sous-espace.

Dans la combinaison ACP+LDA [109, 148], l'ACP est appliquée en premier. Elle permet de réduire les caractéristiques sans tenir compte des étiquettes des classes. LDA est ensuite appliquée pour trouver les composantes qui maximisent la distinction entre plusieurs classes. Enfin, un modèle de classification est utilisé sur ces caractéristiques réduites. Singh et al. [123] ont expérimenté les méthodes ACP, LDA ainsi que la combinaison des deux méthodes de réduction des dimensions, l'ACP et la LDA [109, 148] pour la classification du texte. Sur plusieurs jeux de données, les résultats de Singh et al. [123] ont montré que l'ACP est plus performante que LDA ainsi que la combinaison ACP et LDA.

Une autre possibilité est l'utilisation des algorithmes d'extraction de caractéristiques non-linéaires basés sur l'optimisation de la fonction noyau. Ces algorithmes font référence à une fonction noyau dans d'autres algorithmes et réalisent la transformation d'un problème non-linéaire dans l'espace d'origine en un problème linéaire dans l'espace des caractéristiques [68]. En effet, l'analyse en composantes principales à base de noyau (KPCA : Kernel-Based Principal Component Analysis) [136] consiste à transformer les données d'entrée dans un nouvel espace de caractéristiques par le biais d'une projection non-linéaire, puis à utiliser l'ACP linéaire sur cet espace. Les fonctions de noyau les plus fréquemment utilisées sont les fonctions polynomiales, les fonctions de noyau gaussiennes à base radiale, etc.

4.2.3 Réduction de la dimensionnalité du Multi-GAT

Nous présentons, dans cette section, notre approche de réduction de la dimension des données en entrée du modèle d'inférence. Notre objectif est de limiter les ressources nécessaires pour le chargement des données et le temps de leur traitement tout en améliorant les performances du modèle d'inférence Multi-GAT.

Il est utile de rappeler que le modèle d'inférence de base $Multi - GAT_{DSS}$ que nous réduirons dans cette section pourrait se résumer en plusieurs éléments. Le calcul des caractéristiques textuelles des mots se fait en concaténant les plongements BPEmb des trois premiers sous-mots résultant de la segmentation du mot formant ainsi un vecteur de dimension 300. Le $Multi - GAT_{DSS}$ est multilingue, capable de traiter les documents français et anglais en même temps en employant des modèles BPEmb avec une taille de vocabulaire de 200K suivant l'Algorithme 1. De plus, un vecteur de nature du mot est ajouté au vecteur de plongement textuel. Les caractéristiques textuelles sont enrichies par l'ajout de la position normalisée, l'encodage de la région et le plongement visuel des mots par concaténation formant ainsi un vecteur final de dimension de 337. La matrice d'adjacence modélisant le graphe, est limitée à une dimension de 256 suivant l'Algorithme 2 et veille à inclure tous les voisins des mots du premier niveau.

Nous basons donc notre approche de réduction de la dimensionnalité de $Multi-GAT_{DSS}$ sur chacun des points évoqués ci-dessus. Tout d'abord, nous réduisons la dimension du vecteur de caractéristiques textuelles de 300 à 100 en proposant une fonction simple pour fusionner les plongements des sous-mots. Afin d'améliorer ses performances, le Multi-GAT est en effet transformé en un modèle monolingue. Nous effectuons ensuite une recherche aléatoire pour sélectionner les caractéristiques multimodales des mots, et nous proposons une méthode simple pour les fusionner en utilisant des couches denses. Enfin, nous modifions légèrement l'algorithme de construction des matrices modélisant le graphe, afin d'améliorer encore les performances du modèle.

Calcul des caractéristiques textuelles

Les modèles BPEmb pré-entraînés [54] sont utilisés comme base pour le calcul du plongement textuel des mots. BPEmb offre plusieurs tailles de vocabulaire et dimensions de vecteur qui sont adaptées à l'utilisation sous contraintes de disponibilité de ressources. Après l'application d'une recherche aléatoire sur les différents modèles et vocabulaires de BPEmb, nous adoptons les modèles BPEmb monolingues français et anglais avec le vocabulaire de taille 100 K et le vecteur de plongement de dimension 100 qui restent suffisamment efficaces bien qu'ils consomment moins de mémoire.

Pour chaque mot w dans le DSS dont la segmentation donne un ensemble S_w de n sous-mots s_i, au lieu de concaténer les représentations des trois premiers sous-mots, le vecteur du plongement textuel de w est obtenu à l'aide d'une simple fonction « Moyenne » de tous les "plongements" pré-entraînés de ses sous-mots, où Emb est le vecteur de plongement pré-entraîné du sous-mot.

$$Emb_w = Moyenne(S_w) = \frac{1}{n}\sum_{1}^{n} Emb(s_i) \tag{4.1}$$

Cette représentation permet de prendre en compte de manière égale tous les sous-mots qui composent le mot entier, sans ajout de paddings qui sont insignifiants et n'apportent aucune information sur les caractéristiques du mot.

Sélection et fusion des caractéristiques multimodales

Afin de sélectionner les caractéristiques multimodales les plus pertinentes des mots dans les DSSs, nous procédons à une méthode de sélection par enveloppe qui correspond à une recherche aléatoire sur les différentes modalités en utilisant plusieurs représentations des caractéristiques et en évaluant à chaque fois les performances du Multi-GAT. Dans cette étape, nous éliminons les deux vecteurs correspondant à la nature du mot et à l'encodage de sa région. Nous considérons que ces deux modalités peuvent être indirectement intégrées et apprises à partir des deux vecteurs de plongement textuel et de la position normalisée dans le DSS.

Pour former le vecteur de caractéristiques multimodales, nous regroupons donc les caractéristiques textuelles calculées, la position 2D normalisée (x_1, y_1, x_2, y_2) ainsi que le plongement visuel (obtenu par un ResNet Unet [53, 119] suivi du calcul de la ROI) de la boîte englobante du mot.

Le vecteur final des caractéristiques des mots est obtenu en combinant le vecteur du « plongement textuel » avec la position 2D normalisée encodée à l'aide d'une couche dense (voir la Figure 4.1), dotée d'une fonction d'activation ReLU. Cette couche permet de mieux adapter ces caractéristiques à la tâche de la classification et d'apprendre également les interactions entre les différentes dimensions textuelles et l'encodage de la position. Nous passons ensuite le vecteur du plongement visuel dans une autre couche dense afin de l'adapter à la classification. Enfin, nous concaténons les sorties des deux couches denses pour former le vecteur de caractéristiques final.

Nous appliquons la première couche dense directement sur l'encodage de la position sans aucune couche d'embedding antérieure, ce qui évite d'utiliser davantage de paramètres d'apprentissage. Le résultat de cette approche de sélection et d'extraction de caractéristiques est un vecteur de caractéristique de dimension 128.

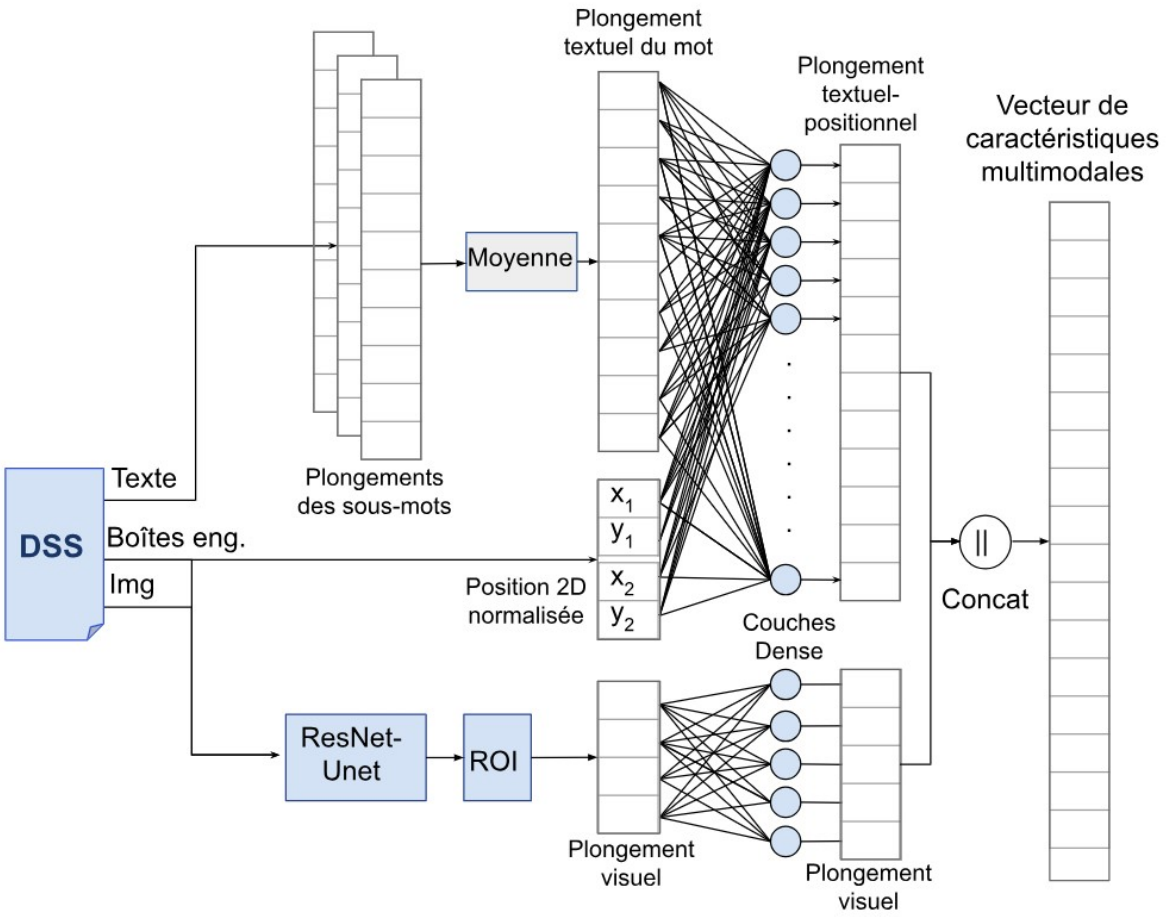


FIGURE 4.1 – Fusion des caractéristiques textuelles et des caractéristiques multimodales pour la formation du vecteur de caractéristiques du mot.

Amélioration de la modélisation des matrices du graphe

Nous améliorons également les performances du modèle en mettant à jour la méthode de la modélisation des matrices représentant le graphe, à savoir la matrice de caractéristiques (X) et la matrice d'adjacence (A), et en apportant une simple modification à l'Algorithme 2.

Le but de cette nouvelle méthode est de conserver les avantages de la première modélisation des matrices en limitant leur dimension à n_max, tout en optimisant la prise en compte des niveaux de voisinage lointains et en préservant, dans la mesure du possible, le voisinage non immédiat des mots dans le graphe.

Dans la méthode décrite par l'Algorithme 2, nous parcourons les mots des documents du corpus et nous remplissons progressivement (avec les mots et leurs voisins) les matrices X et A jusqu'à ce que nous atteignions la taille maximale n_max. Il est ainsi possible de séparer un document sur deux matrices, même s'il contient moins de n_max mots.

Dans la nouvelle modélisation expliquée dans l'Algorithme 3, si un document peut se tenir sur une seule matrice sans avoir à se diviser (il contient un nombre de mots $\leq n\_max$, on lui réserve une nouvelle matrice.

Pour ce faire, on remplit la matrice temporaire, comme indiqué dans l'Algorithme 3, en ajoutant des paddings et on la sauvegarde, puis on affecte le nouveau DSS à une nouvelle matrice. Cela

Algorithme 3 Nouvelle construction des matrices d'adjacence et de caractéristiques pour le $Multi - GAT_{DSS}$.

Entrée : $liste_{DSSs}, n\_max$
Résultat : $liste(X, A, y)$
$X, A, Y \leftarrow$ Null
Pour chaque DSS dans $liste_{DSSs}$ **Faire :**
$X_0 \leftarrow$ Calcul de caractéristiques (DSS)
$A_0 \leftarrow$ Calcul du voisinage (DSS)
$Y_0 \leftarrow$ Classes des mots du DSS
Si $Taille(A_0) \leq n\_max - Taille(A)$ **Alors :**
$X \leftarrow X_0$, $A \leftarrow A_0$, $Y \leftarrow Y_0$
Sinon :
Si $Taille(A_0) \leq n\_max$ **Alors :** ▷ Cette condition correspond à l'amélioration proposée
Ajouter $(X_{tmp}, A_{tmp}, Y_{tmp})$ à (X, A, Y)
Ajouter des Paddings (0) à (X, A, Y)
Ajouter (X,A,Y) à $liste_{DSSs}$
$(X_{tmp}, A_{tmp}, Y_{tmp}) \leftarrow (X_0, A_0, Y_0)$
Sinon :
Pour chaque noeud i dans A_0 **Faire :**
Si Taille(A_{tmp}+ i + Voisins(i) $\leq A_{tmp}$ **Alors :**
Ajouter i et les voisins directs de i à A_{tmp}
Ajouter i et les voisins à X_{tmp}
Ajouter i à Y_{tmp}
Sinon :
Ajouter $(X_{tmp}, A_{tmp}, Y_{tmp})$ à (X, A, Y)
Ajouter des Paddings (0) à (X, A, Y)
Ajouter (X,A,Y) à $liste_{DSSs}$
Fin Si
Fin Pour
Fin Si
Fin Si
Fin Pour
Retourner: $liste_{DSSs}$

permet de préserver autant que possible les relations de voisinage du mot de niveaux supérieurs.

4.2.4 Expérimentations et résultats

Jeux de données

Pour évaluer ce modèle d'inférence, nous utilisons le jeu de donnée SROIE et trois autres jeux de données de factures artificielles : Gen-Invoices, Gen-Invoices-Fr et Gen-Invoices-En. Gen-Invoices est le jeu de données utilisé pour tester le modèle $Multi - GAT_{DSS}$, qui est constitué de 700 DSSs pour l'entraînement, 200 pour la validation et 300 pour le test. Gen-Invoices-Fr et Gen-Invoices-En sont deux jeux de données de factures générées en français et en anglais respectivement. Les deux jeux de données sont divisés comme suit : 1000 factures pour l'entraînement, 200 pour la validation et 300 pour le test.

Choix du modèle de plongement textuel

Afin de choisir le modèle BPEmb à utiliser pour le calcul des caractéristiques textuelles, nous varions les modèles BPEmb et nous évaluons les performances du classifieur Multi-GAT sur les jeux de données de factures monolingues et multilingues. Nous évaluons les modèles avec les vocabulaires de tailles $\geq 50K$, pour une segmentation plus significative, capable de capturer les informations les plus pertinentes du mot. Nous choisissons de concaténer les quatre sous-mots au lieu des trois pour s'assurer de couvrir 99% des mots dans tous les jeux de données choisis.

Jeu de données	Modèle	Vocabulaire	Dimension du vecteur des sous-mots				
			25	50	100	200	300
Gen-Invoices	Monolingue	50K	91.19	91.90	92.47	93.91	93.70
		100K	90.31	92.47	93.22	94.10	93.99
		200K	90.47	91.16	92.59	94.07	94.71
	Multilingue	100K	-	-	-	-	**96.03**
		320K	-	-	-	-	96.00
Gen-Invoices-En	Monolingue	50K	97.53	97.58	97.83	97.74	97.17
		100K	97.70	97.17	**97.85**	**97.86**	97.07
		200K	97.38	97.39	97.54	97.25	97.07
	Multilingue	100K	-	-	-	-	97.82
		320K	-	-	-	-	97.66
Gen-Invoices-Fr	Monolingue	50K	95.47	95.20	96.15	96.15	96.37
		100K	95.20	92.75	**96.40**	95.23	95.91
		200K	95.10	96.21	96.30	95.26	95.47
	Multilingue	100K	-	-	-	-	96.33
		320K	-	-	-	-	95.93

TABLE 4.1 – Score F1 (%) obtenu par le Multi-GAT avec la concaténation des vecteurs des quatre premiers sous-mots.

Comme on peut le voir dans la Table 4.1, les résultats obtenus en utilisant un seul modèle « multilingue » sur Gen-Invoices, qui est un jeu de données mélangé de factures françaises et anglaises, montrent que : les BPEmb multilingues avec le vocabulaire de taille 100K et le vecteur de sous-mots de dimension 300, surpassent les modèles monolingues utilisant l'Algorithme 1. Cependant, cette configuration produit un vecteur final de dimension 1200 et ne permet pas de réduire la dimension du vecteur de caractéristiques de départ.

Pour les jeux de données Gen-Invoices-Fr et Gen-Invoices-En, les résultats obtenus avec les vocabulaires monolingues surpassent les modèles multilingues. Le vocabulaire et la dimension du vecteur de sous-mots qui donnent les meilleurs résultats, sont respectivement 100K et 100. Nous choisissons ainsi de travailler avec les modèles BPEmb monolingues, avec un vocabulaire de 100K et un vecteur de dimension 100.

Extraction des caractéristiques textuelles

Nous évaluons dans cette sous section les méthodes de réduction de dimentionnalité sur le vecteur de plongement textuel, les plus utilisées dans la littérature. La concaténation prend jusqu'à 4 sous-mots au maximum produisant un vecteur de dimension 400. Nous comparons les résultats du vecteur initial avec l'application des algorithmes de l'ACP et du KPCA avec plusieurs noyaux qui réduisent la dimension finale de 400 à 100. Pour l'ACP, nous calculons la

matrice de vecteurs propres sur tout l'ensemble d'entraînement, et ensuite, nous l'appliquons directement sur l'ensemble de test.

Jeu de données	Méthode de réduction par projection des données				
	-	ACP	KPCA (rbf)	KPCA (sigmoid)	KPCA (poly)
Gen-Invoices-En	97.85	97.21	97.81	97.58	**98.17**
Gen-Invoices-Fr	96.40	**96.72**	95.39	95.16	95.96
SROIE	**97.83**	97.25	97.22	97.42	97.43

TABLE 4.2 – Score F1 (%) résultant de la réduction de dimensionnalité en utilisant l'ACP et KPCA.

Pour le KPCA, nous utilisons les 100 premiers documents de chaque jeu de données pour le calcul, car le calcul de certains noyaux sur tous les mots du jeu de données d'entraînement nécessite l'utilisation d'un espace mémoire immense. Cet algorithme transforme les données initiales en leur appliquant le noyau $K(x_a, x_b)$, tel que x_a et x_b sont deux vecteurs de la matrice de caractéristiques, ensuite il applique l'ACP sur le résultat. Nous testons les noyaux rbf (radial basis function), sigmoid et poly qui correspondent aux expressions suivantes :

$$rbf : K(x_a, x_b) = exp(-\sigma||x_a - x_b||^2), \sigma = 1e^{-3}$$

$$sigmoid : K(x_a, x_b) = tanh(\alpha x_a^T x_b + b), \alpha = 1e^{-3}$$

$$poly : K(x_a, x_b) = (x_a^T x_b + b)^d, d = 3$$

Les résultats après application de l'ACP et du KPCA diminuent pour tous les jeux de données à l'exception de KPCA (poly) pour Gen-Invoices-En et ACP pour Gen-Invoices-Fr. La réduction avec KPCA (poly) donne le meilleur résultat de réduction global. Cette perte de performance peut être expliquée par le fait que les paramètres de ACP/KPCA sont calculés sur les données d'entraînement uniquement. Si les données de test sont très différentes de celles d'entraînement, les paramètres de ACP/KPCA calculés peuvent ne pas leur correspondre parfaitement.

On teste aussi d'autres méthodes de réduction de la dimension par fusion de caractéristiques. Nous évaluons différentes méthodes de fusion de vecteurs de sous-mots qui ne nécessitent pas de paramètres d'apprentissage supplémentaires sur les trois jeux de données. Par exemple, Max calcule le maximum (par élément) de tous les vecteurs de sous-mots. Similaire à fastText, dont l'efficacité a été démontrée dans [154] par rapport aux fusions basées sur l'attention, la fonction Sum calcule la somme des vecteurs de sous-mots. Notre fonction Moyenne (présentée dans l'équation 4.1) calcule la moyenne simple sur les vecteurs de sous-mots, tandis que la fonction $Softmax\_mean$ est une moyenne pondérée par la softmax de la longueur des sous-mot par rapport à la longueur du mot. $Softmax\_mean = \sum_1^n Softmax(l_i)Emb_i$, où l_i est la longueur du sous-mot.

TABLE 4.3 – Score F1 (%) obtenu sur les trois jeux de données en utilisant différentes méthodes de fusion de sous-mots.

Jeu de données	Concaténation [8]	Max	Sum [12]	Moyenne	Softmax_mean
SROIE	97.86	**98.14**	98.02	**98.08**	97.95
Gen-Invoices-Fr	96.99	96.99	95.71	**97.40**	97.22
Gen-Invoices-En	98.44	98.10	98.35	**98.44**	98.10

La fonction moyenne obtient les meilleures performances, comme le montre la Table 4.3. Cela peut s'expliquer par le fait que notre fonction Moyenne, contrairement aux autres approches, considère tous les vecteurs de sous-mots de la même manière et n'omet aucune donnée de sous-mots. L'utilisation de la concaténation et de Max peut entraîner une perte de données car Max retient une valeur par dimension sur les vecteurs de sous-mots alors que la concaténation peut ignorer certaines parties du mot s'il est segmenté en plus de quatre sous-mots. La somme et le Softmax_mean sont également moins efficaces que la moyenne. Cela montre que la longueur des sous-mots n'a pas d'impact sur leur importance au sein du mot. Nous pouvons en déduire que chaque sous-mot dans le plongement textuel du mot a une importance similaire et que la « Moyenne » est la fonction qui fusionne le mieux tous les vecteurs de sous-mots dans notre cas. La moyenne surpasse aussi les résultats de la concaténation suivie de ACP/KPCA.

Combinaison de caractéristiques multimodales

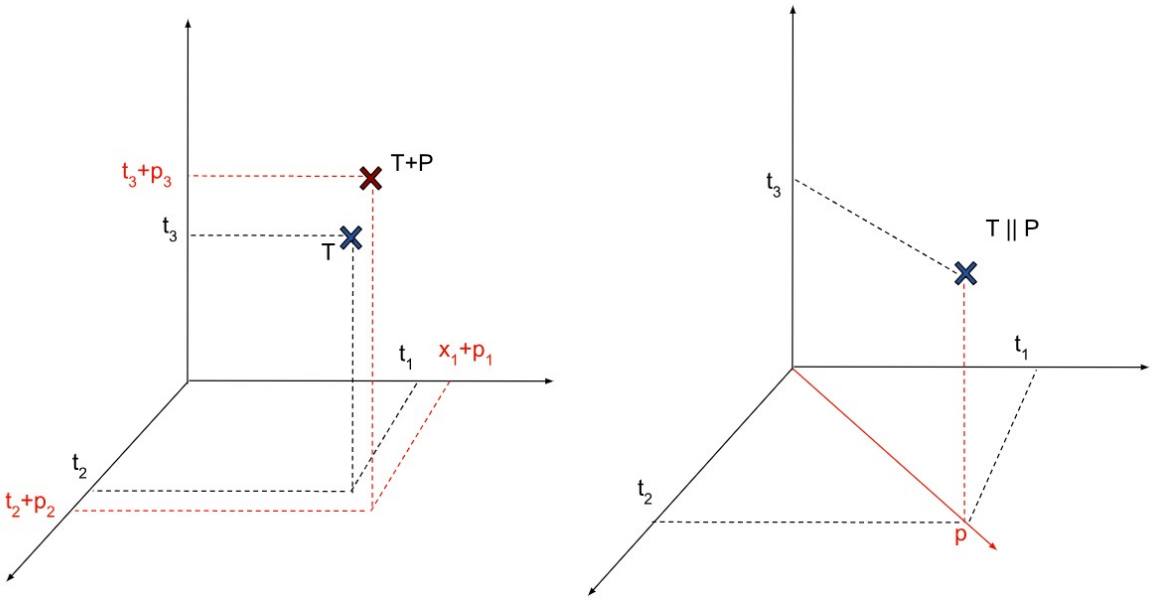
Nous comparons notre méthode de fusion avec d'autres méthodes basées sur les fonctions de fusion par somme et concaténation qui ont été proposées dans les meilleurs systèmes d'EI tels que [8, 41, 84, 134, 135]. Nous testons également différentes représentations de positions tel que Sin qui est un encodage de mise en page utilisé dans [41] basé sur un calcul sinusoïdal, Emb est un encodage de la position suggéré dans [135] qui utilise deux couches d'embedding pour représenter les coordonnées de la boîte englobante du mot, et Enc qui est notre encodage de position normalisé sur une dimension de 4.

Caractéristiques			Fusion	Dim.	SROIE	Gen-Invoices-Fr	Gen-Invoices-En
Texte (T)	Position (P)	Image (I)					
+	Enc	-	T||P	108	98.06	96.26	97.81
+	Sin	-	T||P	200	98.05	96.67	98.23
+	Sin	-	T+P	100	97.88	96.59	98.12
+	Sin	-	T||P	108	97.52	97.19	98.41
+	Emb	-	T||P	200	97.08	96.51	98.53
+	Emb	-	T+P	100	96.94	96.76	98.05
+	Emb	-	T||P	108	97.71	97.31	98.38
+	Emb	+	(T||P)+I	108	96.41	94.92	97.86
+	Emb	+	T||P||I	128	97.75	97.83	98.51
+	Emb	+	Dense(T)||Dense(P) ||Dense(I)	128	98.05	97.37	98.49
+	Enc	+	Dense(T,P,I)	128	98.08	97.35	98.33
+	Enc	+	Dense(T||P)||Dense(I)	128	**98.22**	**98.41**	**99.16**

TABLE 4.4 – Score F1 (%) obtenu sur les trois jeux de données en combinant les différentes caractéristiques à l'aide de différentes méthodes de fusion.

Les résultats de la Table 4.4 montrent que la concaténation des caractéristiques est plus performante que la somme. En effet, l'ajout de la position (P) au texte (T), qu'il s'agisse de Emb ou de Sin, donne de meilleurs résultats que la somme. Il en va de même pour les caractéristiques de l'image (I). Cela tient au fait que le calcul de $T + P$ implique la modification des coordonnées de T le long de chacun des axes de son repère actuel. P ne peut donc plus être séparé ou extrait de T, ce qui peut entraîner une confusion entre différentes paires (T,P) dont la somme est identique. Cependant, concaténer T et P revient à augmenter la taille du référentiel initial en fonction de la

taille de P (si p est sur k positions, on ajoute k à la dimension), donc les caractéristiques restent indépendantes, comme le montre la figure 4.2. Nous notons également que la concaténation d'un vecteur de position de dimension 8 donne de meilleurs résultats qu'un vecteur de dimension 100. Un vecteur de dimension 8 est suffisant pour capturer des informations utiles sur la position.



(a) Somme de deux modalités de caractéristiques T et P.

(b) Concaténation de deux modalités de caractéristiques T et P.

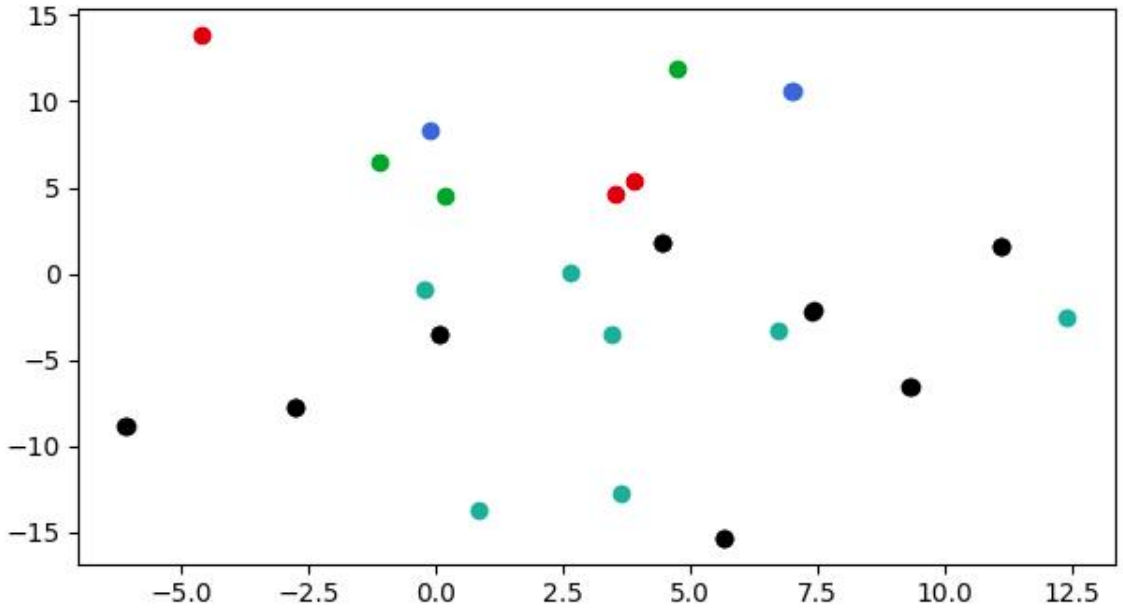
FIGURE 4.2 – Comparaison entre la somme et la concaténation des caractéristiques.

La concaténation des caractéristiques du texte (dimension 100), des caractéristiques de position obtenues par Emb (dimension 8) et de l'image (dimension 20) constitue la meilleure configuration, comparée aux autres utilisant des sommes ou une combinaison de la somme et de la concaténation.

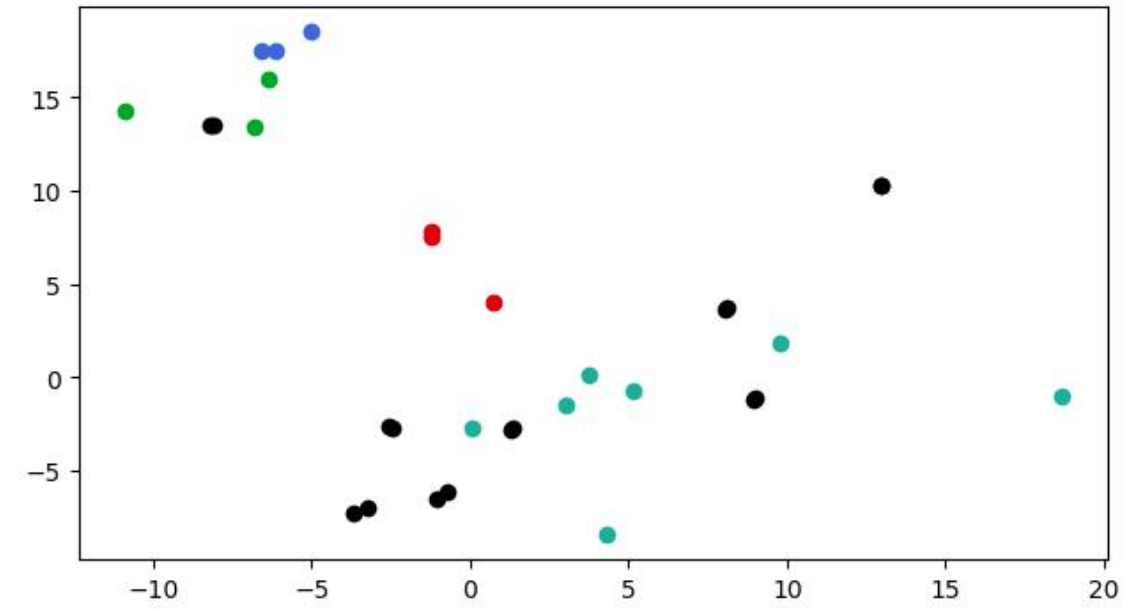
Afin d'améliorer les performances de cette méthode de fusion et de réduire au mieux les paramètres des couches de plongement de position utilisées, nous expérimentons d'autres fonctions de fusion basées sur l'application de la concaténation et des couches denses sur les différentes caractéristiques.

Comme le montre la Table 4.4, le meilleur résultat est obtenu par le vecteur multimodal de dimension 128, en concaténant les sorties de deux couches denses appliquées aux trois caractéristiques multimodales (texte, encodage de la position (Enc) et image). Cette méthode de fusion est plus efficace que la simple concaténation, car elle nous permet d'apprendre les interactions entre les caractéristiques textuelles et de position, et de mieux les adapter à la tâche de classification.

Nous avons également comparé la dispersion des vecteurs de caractéristiques de cinq classes dans trois factures en utilisant une simple concaténation des caractéristiques multimodales et en y ajoutant les couches denses comme nous l'avons proposé. Comme le montre la Figure 4.3, l'ajout de couches denses (4.3b) réduit la dispersion entre les vecteurs appartenant aux mêmes classes, réduisant ainsi la distance intra-classe. Nous remarquons également que les distances inter-classes augmentent et que la séparation entre les classes est mieux visualisée après l'ajout de couches denses (Figure 4.3b). Par conséquent, l'utilisation de ces vecteurs comme entrée du classifieur, permet d'optimiser davantage la tâche de classification.



(a) Dispersion des vecteurs sans couches denses.



(b) Dispersion des vecteurs avec couches denses.

FIGURE 4.3 – Comparaison des dispersions des vecteurs de caractéristiques des mots de cinq classes dans trois documents extraits de Gen-Invoices-En avec et sans l'application des couches denses pour leur fusion multimodale.

Matrice d'adjacence

Pour étudier l'effet de la matrice d'adjacence proposée, nous évaluons le modèle en utilisant deux matrices d'adjacence différentes sur les jeux de données Gen-Invoices-En et Gen-Invoices-Fr.

La matrice Padding_adj modélise un document à la fois, avec un nombre maximum de nœuds fixé à 512 pour s'adapter à tous les documents dans les jeux de données. Des nœuds padding sont utilisés pour remplir le graphe si le nombre de mots du document est inférieur à 512. Adj_256 correspond à notre matrice d'adjacence avec $n\_max = 256$ qui correspond au nombre moyen de mots du DSS dans les jeux de données.

TABLE 4.5 – Score F1 (%), nombre d'époques et temps de traitement d'un DSS (en millisecondes) obtenus sur Gen-Invoices-En et Gen-Invoices-Fr en utilisant différentes matrices d'adjacence.

	Gen-Invoices-En		Gen-Invoices-Fr	
Matrice d'adjacence	Padding_adj	Adj_256	Padding_adj	Adj_256
Score F1	98.47	99.16	98.21	98.41
Nombre d'époques	250	190	250	226
Temps de traitement (ms)	431	369	400	373

Les résultats de la Table 4.5 montrent que la matrice d'adjacence que nous proposons fournit les meilleurs résultats tout en limitant le nombre d'étapes d'apprentissage et le temps de traitement d'un seul DSS. La matrice d'adjacence proposée rend le modèle plus efficace avec moins de ressources mémoire.

Résultats globaux

Nous comparons les résultats et la complexité de notre système avec ceux des modèles les plus performants de la littérature sur le jeu de données SROIE.

TABLE 4.6 – Comparaison du score F1 (%) entre notre système et les systèmes de la littérature (M signifie million).

Système	# Paramètres	Score F1 (%)
TRIE++[19]	-	**98.37**
LAMBERT[41]	125M	98.17
LayoutLMv2(L)[134]	426M	97.81
LayoutLMv2(B)[134]	200M	96.25
StrucText[84]	107M	96.88
ViBERTGrid[88]	157M	96.40
TRIE[145]	–	96.18
PICK[139]	–	96.12
LayoutLM(L)[135]	343M	95.24
LayoutLM(B)[135]	113M	94.38
BERT(B)[36]	340M	92
Multi-GAT	41M	**98.22**

Comme le montre la Table 4.6, notre système obtient des résultats compétitifs par rapport aux autres systèmes avec une complexité bien moindre. Nous rappelons que ce modèle évalué à cette étape est un modèle purement supervisé construit sans étape de pré-entraînement, contrairement aux autres systèmes basés sur les transformers.

Analyse des erreurs

Le jeu de données SROIE donne lieu à plusieurs erreurs de classification, où des entités comme « Total », « Date », etc., se retrouvent classées à tort (des faux négatifs) comme « undefined », (voir quelques exemples dans la Figure 4.4).

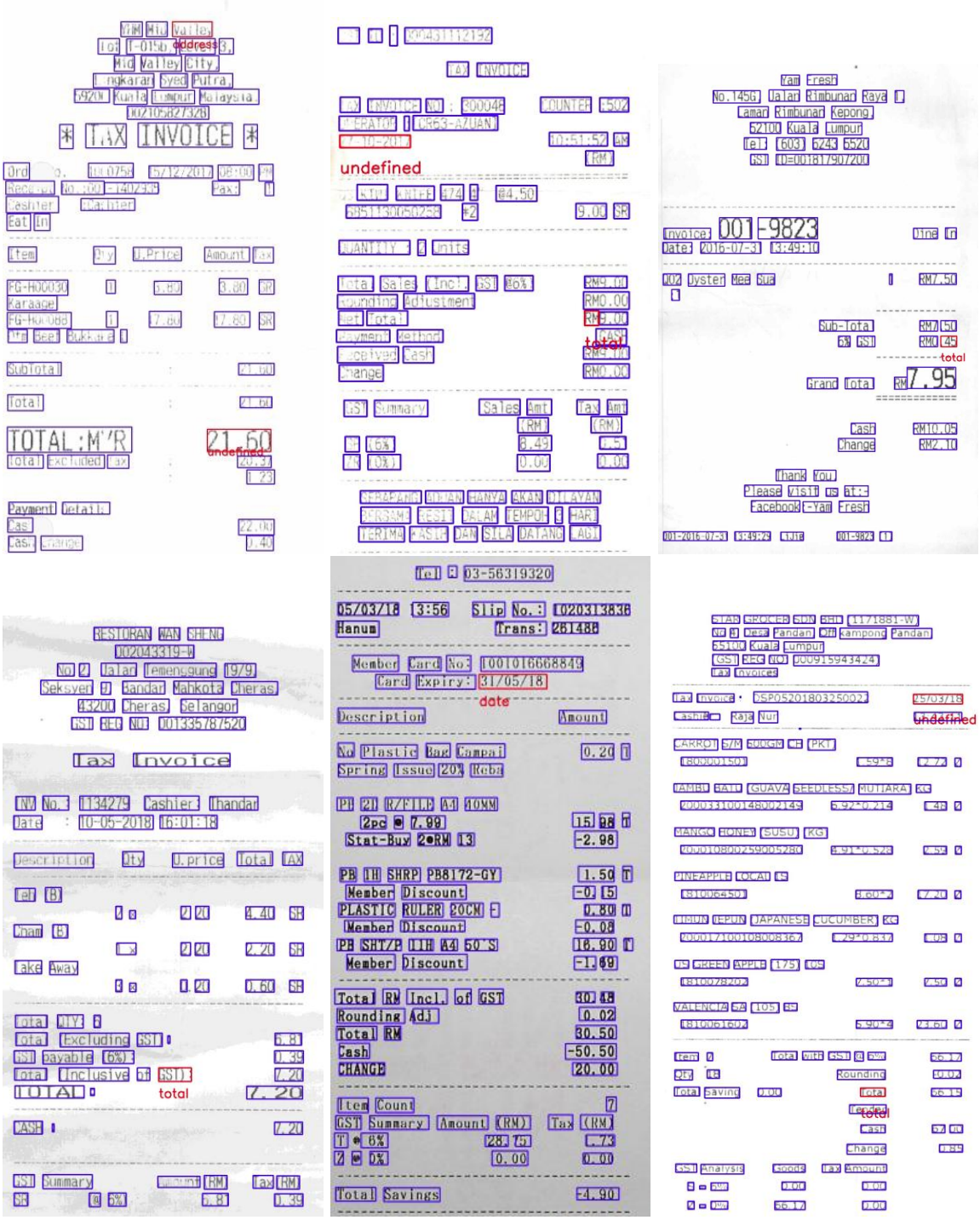


FIGURE 4.4 – Exemples d'erreurs engendrées sur SROIE (les mots encadrés et étiquetés en rouge).

Ces entités sont souvent confondues avec d'autres mots qui ont des propriétés similaires, mais qui appartiennent généralement à la classe « undefined », comme les dates d'impression, les sous-totaux, etc.

Des faux positifs sont aussi observés où des mots sont prédits comme « Total » alors qu'il s'agit de mots appartenant à la classe « undefined ». Ces faux positifs se trouvent dans des positions très proches du vrai « Total » et ils sont entourés par les mots-clés de cette classe. Ils peuvent correspondre aux sous-totaux par exemple. La Figure 4.4 montre également des faux positifs de la classe « Date » : la date du ticket de caisse est parfois confondue avec d'autres dates, telles que la date d'expiration, la date d'impression, etc. qui ont le même format, mais appartiennent à la classe « undefined ». Cette erreur est due au fait que les contenus de ces dates sont très semblables, se trouvent souvent dans la même région du document et que la « Date » n'est pas toujours précédée d'un mot-clé introductif. Cela peut rendre la distinction de l'entité correcte plus difficile.

Un autre type d'erreur observé dans les résultats est la confusion entre certains mots appartenant aux classes « Company » et « Address ». Cette confusion vient du fait que ces deux entités se suivent dans les documents, ne sont pas séparées visuellement et peuvent également contenir des mots identiques.

Certaines de ces erreurs peuvent être corrigées facilement lors du post-traitement. Les faux positifs de la classe « Total » peuvent être éliminés si le mot prédit est alphabétique ou alphanumérique. Les mots confondus entre les classes « Company » et « Address » peuvent aussi être corrigés lorsque le mot se trouve entre deux mots bien classés appartenant à la même entité.

Sensibilité du modèle aux erreurs de l'OCR

Pour montrer l'efficacité du Multi-GAT face à des erreurs possibles d'OCR, nous avons simulé ces erreurs sur la base de test du jeu de données Gen-Invoices-En. Plus précisément, nous avons introduit trois types d'erreurs sur les mots des DSSs : la suppression, l'ajout et la substitution de caractères dans des positions aléatoires dans les mots. Nous avons fait varier le pourcentage des mots sur lesquels nous avons introduit ces erreurs.

% des erreurs d'OCR	0	2	3	4	5	10
Score F1 (%)	99.16	98.07	98.70	96.82	96.35	93.64

TABLE 4.7 – Résultats (Score F1 %) du Multi-GAT sur le jeu de données Gen-Invoices-En en variant le pourcentage des erreurs d'OCR dans l'ensemble de test.

La Table 4.7 montre que les performances du modèle diminuent au fur et à mesure que le pourcentage d'erreurs d'OCR augmente dans l'ensemble de test. Néanmoins le pourcentage de la perte de performance reste toujours inférieur au pourcentage des erreurs introduites. Cela montre que le modèle réussit à bien classer plusieurs mots en dépit des erreurs d'OCR introduites.

Pour mieux visualiser l'impact des erreurs sur la classification, nous analysons quelques résultats où le modèle est insensible aux erreurs et d'autres où il est affecté par celles-ci. Dans les exemples qui suivent, les mots encadrés en vert sont les mots dans lesquels nous avons introduit des erreurs aléatoires, ceux encadrés en bleu sont les mots bien classés par le modèle et ceux encadrés et etiquetés en rouge sont les mots mal classés par le modèle.

Le Multi-GAT parvient à classer efficacement des mots erronés en fonction de leur type (mot-clé, information ou autre), de la nature de leurs caractères (alphabétiques, numériques ou spéciaux) et du type d'erreur introduite (suppression d'un caractère, ajout, etc.).

— Les erreurs dans les mots-clés peuvent ne pas avoir d'impact significatif sur la classification, en particulier lorsque les mots-clés sont constitués de plusieurs mots (tels que « Shipping Address » et « Tax Amount » dans l'exemple de la Figure 4.5). En outre, le voisinage local non immédiat du mot peut également compenser l'effet de ces erreurs.

FIGURE 4.5 – Exemples d'introduction d'erreurs dans les mots-clés.

— Si le mot erroné appartient à l'information et non aux mots-clés introductifs, il n'affecte habituellement pas la prédiction de sa classe et celles de ses voisins (voir les exemples dans la Figure 4.6).

FIGURE 4.6 – Exemples d'introduction d'erreurs dans les informations.

— Si le mot est segmenté en plusieurs sous-mots, la fusion des caractéristiques textuelles permet de le classer correctement (voir les exemples de la Figure 4.7).

FIGURE 4.7 – Exemples d'introduction d'erreurs dans des mots segmentés en plusieurs sous-mots.

— Lorsque l'erreur sur un champ numérique (exemples dans la Figure 4.8) est de type suppression d'un caractère ou substitution d'un caractère par un autre caractère numérique, alors le mot garde toujours sa nature numérique, ce qui est habituellement un des indicateurs les plus importants pour sa classification.

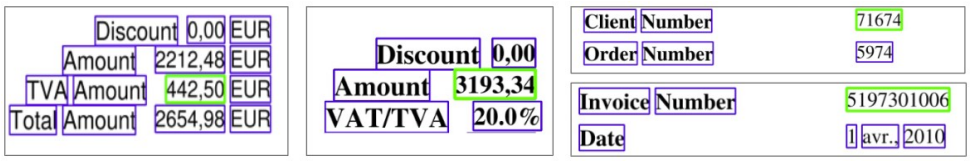


FIGURE 4.8 – Exemples de suppression de caractères dans les champs numériques.

Toutefois, les résultats de la classification du Multi-GAT peuvent parfois être affectés par ces erreurs. Nous énumérons quelques exemples dans ce qui suit.

— Le remplacement du caractère d'un champ numérique (comme les totaux, les prix unitaires, etc.) par un caractère spécial ou alphabétique, ou son ajout au mot, peut modifier sa nature et ses caractéristiques, ce qui affecte sa classification ainsi que celle de ses voisins (voir les exemples illustrés dans la Figure 4.9).

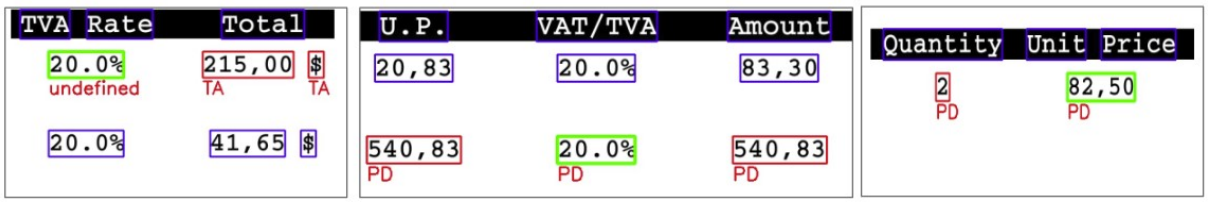


FIGURE 4.9 – Exemples d'erreurs de classification dues au changement de la nature des champs numériques.

— Les erreurs introduites dans les mots-clés, comme le montrent les exemples dans la Figure 4.10, peuvent aussi affecter les résultats de la classification des informations qu'ils accompagnent. Cela se produit lorsque le mot erroné est le seul ou le principal mot-clé introduisant l'information. Cela peut également survenir lorsqu'il est séparé de son bloc d'information de telle sorte que le contexte local dans le bloc ne peut plus compenser l'effet de l'erreur introduite.

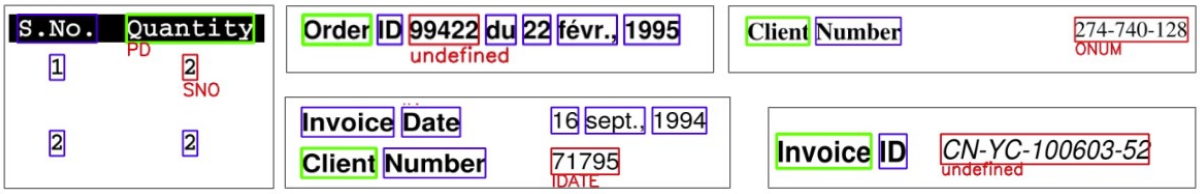


FIGURE 4.10 – Exemples d'erreurs de classification dues à des mots-clés erronés.

Nous analysons également les performances de notre modèle en le comparant à un modèle entraîné sur une base de données augmentée avec des erreurs d'OCR incorporées, qu'on note $Multi - GAT_{Augmented}$ (voir la Table 4.8).

Erreurs d'OCR	Multi-GAT	$Multi - GAT_{Augmented}$
✗	99.16	99.03
✓	96.35	96.01

TABLE 4.8 – Résultats (Score F1 (%)) obtenus sur Gen-Invoices-En par le Multi-GAT et le $Muli - GAT_{Augmented}$ en introduisant des erreurs d'OCR (sur 5% des mots).

On remarque que l'intégration de ces erreurs lors du processus d'apprentissage du modèle n'améliore pas ses performances sur l'ensemble de test contenant des erreurs d'OCR. En fait, cela dégrade les performances du modèle sur les données saines et sur celles qui contiennent des erreurs. Cette approche d'augmentation des données peut ainsi biaiser les résultats du modèle.

L'approche optimale consiste à conserver notre système de base et à utiliser des techniques de pré-traitement des documents qui corrigent les erreurs d'OCR lors de la mise en œuvre de ce modèle sur des données réelles susceptibles de présenter des erreurs.

4.2.5 Synthèse

Dans cette section, nous avons présenté un modèle Multi-GAT à ressources réduites. Nous avons proposé une méthode simple de fusion des vecteurs de sous-mots pré-entraînés pour reconstruire les vecteurs de mots en calculant simplement leur moyenne. Cette méthode ne requiert pas de paramètres d'apprentissage supplémentaires et permet de prendre en compte tous les sous-mots de manière identique. Le vecteur de caractéristiques multimodales final est obtenu à l'aide d'une méthode de fusion simple en concaténant les sorties de deux couches denses appliquées au plongement textuel, positionnel et visuel. L'utilisation de ces deux couches denses a permis de réduire la dispersion des vecteurs de caractéristiques des mots en diminuant les distances intra-classe entre les mots appartenant à la même classe et en augmentant les distances inter-classes. La matrice d'adjacence que nous avons proposée évite les nœuds de padding non informatifs et permet d'obtenir de meilleures performances avec moins de ressources. Le Multi-GAT est insensible à certaines erreurs d'OCR. Cependant, il doit être utilisé en association avec un OCR performant ou un prétraitement sur les jeux de données réels susceptibles de générer des erreurs d'OCR afin de garantir de meilleures performances.

4.3 Adaptation du modèle d'inférence au chiffrement homomorphe

4.3.1 Introduction

Les modèles d'inférence d'EI requièrent habituellement d'être chiffrés afin d'assurer la confidentialité des données des utilisateurs. Cela implique l'utilisation de protocoles de chiffrement homomorphes qui permettent de manipuler directement des données chiffrées sans avoir à les déchiffrer au préalable. Afin de préparer notre modèle au chiffrement, nous effectuons une approximation polynomiale des opérations non homomorphes dans le modèle d'inférence Multi-GAT.

Le déploiement de ces modèles d'EI implique deux acteurs : un client disposant d'une capacité de calcul limitée et des documents à traiter, et un serveur distant très performant dans lequel se loge le modèle d'EI. Le client doit transférer les documents au serveur qui exécute le modèle d'EI et renvoie ensuite les résultats de l'extraction au client. Toutefois, ce schéma peut poser un problème de confidentialité. Le serveur distant est généralement considéré comme peu fiable par le client. Ce dernier ne souhaite donc pas que le serveur ait accès à ses données confidentielles brutes. Ce type de confidentialité peut être garanti à l'aide du chiffrement homomorphe (Homomorphic Encryption (HE)).

Un système de chiffrement homomorphe est un système de chiffrement qui permet d'effectuer des calculs sur des données chiffrées sans devoir les déchiffrer au préalable. Cela signifie que le client peut confier les calculs à un serveur externe, sans que ses données ou les résultats bruts soient accessibles par ce dernier. Le serveur renvoie le résultat au client qui est le seul à pouvoir le déchiffrer.

Il existe plusieurs protocoles de chiffrement homomorphe, toutefois, étant donné que les modèles d'EI opèrent principalement sur des vecteurs de nombres réels, nous nous intéresserons dans ce qui suit, au protocole CKKS [20] qui a la capacité de traiter les nombres réels.

Dans cette section, nous présenterons d'abord l'état de l'art sur les méthodes d'adaptation des opérations neuronales au chiffrement homomorphe, puis nous expliquerons l'approche proposée.

4.3.2 Le protocole CKKS

Le schéma Cheon-Kim-Kim-Song (CKKS) [20] a été introduit en 2017 et est devenu l'un des schémas d'HE les plus largement utilisés. L'une des caractéristiques distinctives du CKKS est qu'il fonctionne avec une arithmétique approximative. Cela le distingue des autres systèmes d'HE tels que BGV [13] et FV [39] qui utilisent l'arithmétique exacte, ainsi que TFHE [24], conçu pour les opérations binaires. Autrement dit, CKKS a la capacité de travailler avec des nombres réels, alors que les autres systèmes sont limités au calcul des nombres entiers. Le CKKS implique de travailler avec des nombres complexes et des polynômes dans un anneau quotient. Une transformation est nécessaire pour passer de l'espace des nombres complexes à l'espace des polynômes. CKKS permet d'effectuer plusieurs opérations telles que l'addition, la multiplication et la rotation.

L'un des principaux obstacles aux opérations CKKS est l'erreur d'approximation induite qui survient lors du chiffrement et augmente lors des calculs successifs. Une opération de remise à l'échelle ou de bootstrapping est proposée après un nombre prédéfini de niveaux d'opérations de multiplication.

4.3.3 État de l'art sur l'approximation polynomiale

La mise en œuvre de modèles neuronaux homomorphes implique l'application de l'approximation sur les opérations non-linéaires en les transformant en d'autres fonctions plus compatibles

avec le chiffrement homomorphe. Afin d'y parvenir, plusieurs approches récentes ont proposé différentes approximations des fonctions d'activation et des couches non-linéaires dans les réseaux neuronaux profonds. Les approches d'apprentissage automatique préservant la confidentialité (PPML : Privacy-Preserving Machine Learning), proposées dans la littérature, peuvent être classées en approches interactives et non interactives [34]. Les approches interactives tolèrent l'utilisation d'opérations non homomorphes, mais nécessitent une connexion entre le client et le serveur, ce qui implique l'exécution de tâches de calcul en ligne. Les approches non interactives ne nécessitent qu'une seule transmission de données au départ par le client, et tous les calculs sont effectués sur le serveur. Cela est rendu possible grâce aux approximations polynomiales de toutes les opérations non-linéaires figurant dans le modèle neuronal, ce qui rend le modèle homomorphe. Dans ce qui suit, nous présentons différentes approximations proposées dans la littérature pour certaines opérations non-linéaires, comme les fonctions d'activation qu'on retrouve dans les modèles neuronaux.

Approximation de la ReLU

La fonction d'activation ReLU est la fonction la plus approximée dans la littérature. Elle est définie par $ReLU(x) = max(0, x)$. Les auteurs de [43, 45, 91, 103] remplacent la fonction ReLU par une simple fonction carrée. Bien que la fonction carrée soit une fonction simple et fonctionne correctement pour des modèles avec un petit nombre de couches et sur des jeux de données simples, elle peut entraîner une forte dégradation des performances des réseaux neuronaux de plus grande profondeur, étant donné qu'après chaque approximation, les résultats s'éloignent de plus en plus de la fonction ReLU de base. [5].

D'autres méthodes tentent de minimiser la différence entre la fonction ReLU et l'approximation polynomiale dans un intervalle $[a, b]$ et produisent des polynômes avec des coefficients fixes qui ne changent pas durant l'entraînement. Les auteurs dans [22, 64, 128, 150] proposent des polynômes de différents degrés en utilisant la méthode des moindres carrés [33] pour calculer les coefficients polynomiaux. Considérant que f correspond à la ReLU, l'approximation polynomiale de f sur l'intervalle $[a, b]$ en utilisant la méthode des moindres carrés, consiste à trouver le polynôme p_n de degré n, qui minimise la fonction de coût $\int_a^b (f(x) - p_n(x))^2 dx$. Les polynômes proposés dans [22, 64, 128] comprennent des coefficients réels tandis que Zheng et al. [150] utilisent des coefficients dans le domaine complexe. Chiang et al. [22] et Zheng et al. [150] proposent des polynômes de degré faibles en utilisant un calcul des moindres carrés basé sur les dérivées [22]. La fonction de coût à minimiser dans cette méthode est définie par l'expression : $\int_a^b (f(x) - p_n(x))^2 + (f'(x) - p'_n(x))^2 dx$. Ali et al. [5] ont également proposé un polynôme de degré 2 avec des coefficients fixes. En revanche, ils utilisent l'algorithme d'optimisation minimax qui a comme objectif de minimiser la valeur $max_{a \leq x \leq b} |f(x) - p_n(x)|$.

Baruch et al. [7] utilisent un polynôme du second degré avec deux coefficients entraînables (a et b) à apprendre individuellement pour chaque couche pendant le processus d'entraînement. Pour apprendre ces deux coefficients, ils utilisent une approche de transition douce. Ils entraînent d'abord le modèle avec des couches d'activation ReLU pour les premières époques, ensuite ils passent progressivement des fonctions ReLU aux fonctions d'activation polynomiales. Pour ce faire, ils définissent une fonction d'activation pondérée avec un paramètre λ_e différent à chaque époque comme $weighted\_act_{\lambda_e}(x) = (1 - \lambda_e) * ReLU(x) + \lambda_e * p_n(x)$. Enfin, ils continuent d'entraîner le modèle en utilisant uniquement les approximations $p_n(x)$. De même, Qian et al. [114] utilisent un ensemble de polynômes d'Hermite de degré faible entraînables. La fonction d'activation qu'ils proposent est une combinaison linéaire de polynômes d'Hermite (appelée LotHps). Les polynômes d'Hermite sont calculés suivant l'expression : $h_0(x) = 1, h_1(x) = 2x$, et

$h_{n+1}(x) = 2xh_n(x) - 2nh_{n-1}(x)$ pour $n \geq 2$. Afin de maintenir une faible profondeur de multiplication, ils utilisent uniquement les trois premiers termes $h_0(x)$, $h_1(x)$ et $h_2(x)$. La fonction LotHps est alors définie par $LotHps(x) = w_0h_0(x) + w_1h_1(x) + w_2h_2(x)$ où w_0, w_1 et w_2 sont trois paramètres à apprendre par le réseau neuronal. Pour obtenir le modèle final contenant les LotHps, un modèle basé sur la ReLU est utilisé pour contrôler son entraînement. Une fonction de coût pondérée est introduite à cet effet par un paramètre réglable à chaque étape d'apprentissage. Il inclut la différence entre les sorties des couches d'activation du modèle basé sur les LotHps et du modèle basé sur la ReLU, ainsi que la différence entre les sorties des couches finales de ces deux modèles.

Approximation de la Softmax

La fonction Softmax est aussi une des fonctions étudiée dans la littérature. Elle est utilisée particulièrement dans le calcul du mécanisme d'attention et définie par l'expression suivante : $Softmax(x) = \frac{exp(x_i)}{\sum_{k \in N_i} exp(x_k)}$.

Afin d'approximer cette fonction, certaines méthodes proposent des fonctions non-linéaires qui nécessitent des connexions avec la partie détenteur des données et l'utilisation du protocole MPC (secure Multi-Party Computation) et d'autres utilisent des approximations polynomiales qui ne font pas appel au protocole MPC.

Dans la première catégorie, Mohassel et al. [103] remplacent la Softmax par une ReLU-Softmax, en modifiant simplement l'exponentielle par une fonction ReLU dans la formule de la Softmax. Chen et al. [17] proposent une autre formule basée sur la fonction ReLU et un réseau neuronal linéaire à trois couches (T), donnant lieu à l'estimation de la Softmax par la fonction S, telle que $S(x_i) = x_i * T(\sum_j ReLU(((x_j)/2 + 1)^3))$. Ici le T est utilisé pour estimer l'opération de la division dans la Softmax. Li et al. [82] proposent la fonction 2Quad où l'exponentielle, dans la formule Softmax, est remplacée par $(x + c)^2$. Toutes ces estimations ont comme objectif de réduire le temps d'inférence lors du calcul en ligne de la Softmax. Li et al. [82] ont montré que leur proposition entraîne un temps d'inférence plus court que celui obtenu avec l'estimation ReLUSoftmax [103]

Dans la deuxième catégorie, Ali et al. [4] remplacent la Softmax par une fonction sigmoïde et l'approximent ensuite par le polynôme de degré 3 proposé par Kim et al. [74]. Chiang [23] remplace également la Softmax par la fonction sigmoïde et utilise la méthode des moindres carrés pour obtenir un polynôme de degré 11, ce dernier étant ensuite approximé par un polynôme de degré 3. Al Badawi et al. [3] remplacent la Softmax par un polynôme de degré 2 en utilisant l'algorithme d'approximation Minimax [99].

Les auteurs de [31, 66] proposent l'approximation de toute fonction non-linéaire par des polynômes de différents degrés (2, 3, 5 ou 7) avec des coefficients entraînables.

Approximation de la normalisation

D'autres travaux ont proposé des approximations pour les couches de normalisation et de normalisation par batch. L'expression de la normalisation de base correspond à : $y = \frac{x-\mu}{\sqrt{\sigma+\epsilon}} * \gamma + \beta$, où μ désigne l'espérance et σ l'écart type calculés sur les données d'entrée, tandis que γ et β correspondent à des paramètres de mise à l'échelle et d'ajustement du résultat.

Chen et al. dans [17] ont remplacé la couche de normalisation par la formule $x * \gamma + \beta$. Pour éviter l'introduction de nouveaux paramètres, ils éliminent l'espérance, l'écart-type et l'opération de division et ne conservent que les paramètres γ et β, à apprendre par le réseau de neurones profond, dans l'expression de la normalisation. Ils considèrent que cette simple formule linéaire est

suffisante pour les valeurs avec un faible intervalle de biais. Les auteurs de [62, 86, 95] reformulent la couche de normalisation par batch en proposant une approximation qui pourrait être allégée en re-paramétrant les poids et les biais des couches précédant la couche de normalisation. Une fois le réseau entraîné, les valeurs de la moyenne et de la variance sont fixées pendant la phase d'inférence [63]. Pour ce faire, des estimations non biaisées de la moyenne et de la variance sont calculées pour tous les batchs. Les coefficients de normalisation sont appris à partir des données d'entraînement du serveur et non des données d'entrée du client.

4.3.4 Approximation polynomiale des fonctions non-linéaires dans le Multi-GAT

Afin de garantir la confidentialité des données, le traitement d'un DSS par le modèle d'inférence se fait principalement en calculant les caractéristiques par le client ou le détenteur des données, puis en les chiffrant et en les envoyant au serveur qui applique le modèle Multi-GAT homomorphe ou partiellement homomorphe pour analyser le document et enfin en renvoyant les résultats. Le schéma de cette évaluation est représenté dans la Figure 4.11.

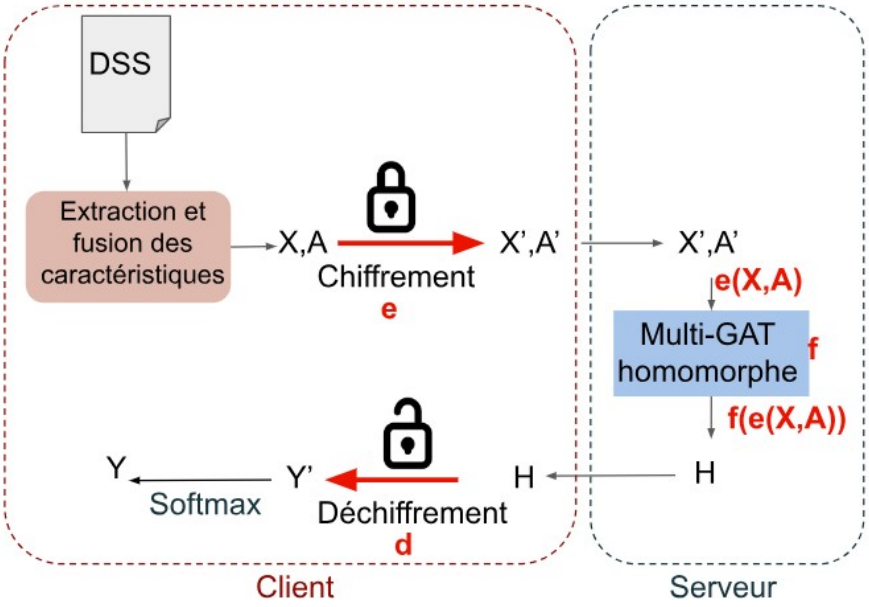


FIGURE 4.11 – Schéma homomorphe de traitement d'un DSS par le Multi-GAT.

On peut effectuer des calculs directement sur des données chiffrées si on a un homomorphisme de chiffrement e, un homomorphisme de déchiffrement d tels que $d(e(x)) = x$ et une fonction f qui est une composition d'additions et de multiplications. Un homomorphisme h étant une fonction qui satisfait les deux propriétés : $h(x + y) = h(x) + h(y)$ et $h(x * y) = h(x) * h(y)$.

Les fonctions non-linéaires dans le Multi-GAT

À titre de rappel, le mécanisme d'attention à k têtes d'attention, calculé par le modèle de base Multi-GAT pour un nœud i dans le graphe, correspond à :

$$\overrightarrow{h'}_i = f_{k\in[1,K]}\sigma(\sum_{j\in N_i} \alpha_{ij}^k W^k \overrightarrow{h}_j) \tag{4.2}$$

où f représente la concaténation $\|$ des résultats des K têtes d'attention pour toutes les couches GAT à l'exception de la couche de sortie, pour laquelle elle représente la moyenne. σ est la fonction d'activation ReLU ou Softmax (dans la couche de sortie).
Les coefficients d'attention α_{ij}^k correspondent à :

$$\alpha_{ij} = Softmax_j(e_{ij}) = \frac{exp(LeakyReLU(\overrightarrow{a}^T[W\overrightarrow{h}_i||W\overrightarrow{h}_j]))}{\sum_{k\in N_i} exp(LeakyReLU(\overrightarrow{a}^T[W\overrightarrow{h}_i||W\overrightarrow{h}_k]))} \tag{4.3}$$

où $\overrightarrow{a}^T$ est la matrice de poids d'attention apprise.

L'ensemble des opérations non-linéaires utilisées dans ce calcul, qui doivent être approximées pour former un Multi-GAT adapté au chiffrement homomorphe, sont les suivantes :

— La fonction d'activation ReLU appliquée à chaque sortie GAT

— La LeakyReLU et la Softmax dans le calcul de l'attention

— La couche de normalisation que nous ajoutons entre les couches GAT afin de mieux stabiliser le processus d'apprentissage.

La première fonction à approximer et la plus étudiée, est la fonction ReLU.

$$ReLU(x) = \begin{cases} x & \text{si } x \geq 0 \\ 0 & \text{sinon.} \end{cases} \tag{4.4}$$

Cette fonction peut être aussi exprimée par $ReLU(x) = max(0, x)$. La condition ou le calcul du max ne sont pas supportés par le chiffrement homomorphe.

La ReLU est légèrement modifiée pour former la fonction LeakyReLU définie comme dans l'équation (4.5).

$$LeakyReLU(x) = \begin{cases} x & \text{si } x \geq 0 \\ \alpha x & \text{sinon.} \end{cases} \tag{4.5}$$

La fonction d'activation Softmax (voir eq. 4.6) contient l'opération exponentielle et l'opération de division qui ne sont pas homomorphes et qui nécessitent donc d'être approximées.

$$Softmax(x) = \frac{exp(x_i)}{\sum_{k\in N_i} exp(x_k)} \tag{4.6}$$

La couche de normalisation

Afin d'assurer plus de stabilité dans les calculs au sein du Multi-GAT, nous intégrons des couches de normalisation entre les couches GAT.

La couche de normalisation de base est définie comme suit :

$$y = \frac{x - \mu}{\sqrt{\sigma + \epsilon}} * \gamma + \beta \tag{4.7}$$

où $\mu = \frac{1}{N}\sum_{i\in N} x_i$ et $\sigma = \frac{1}{N}\sum_{i\in N}(x_i - \mu)^2$

Cette couche contient un calcul de la racine carrée et une opération de division qui ne sont pas homomorphes.

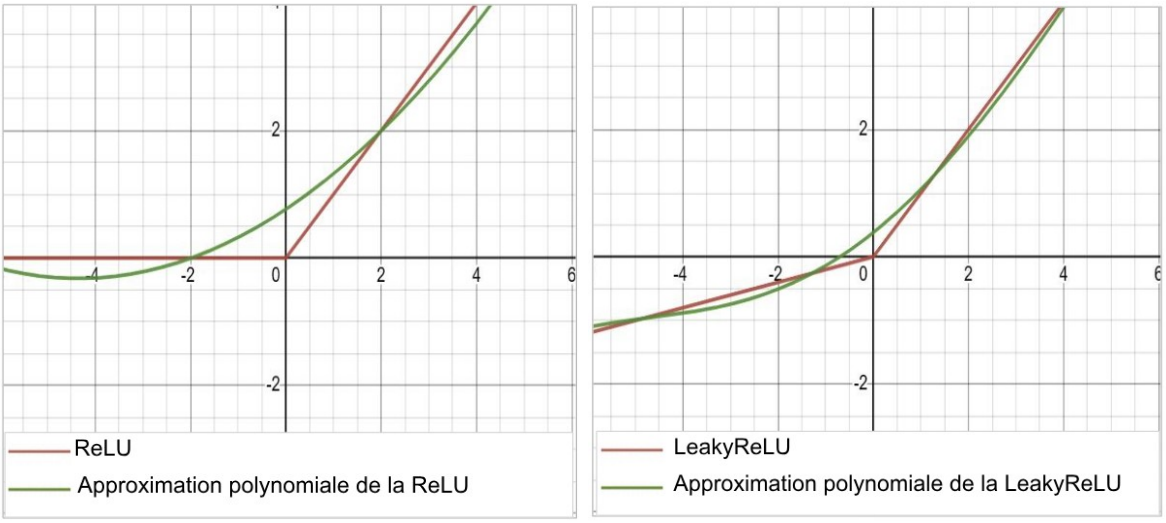
Approximation des fonctions d'activation

Nous remplaçons les fonctions d'activation, à savoir : la ReLU, la LeakyReLU et la Softmax par des polynômes de faible degré, comme indiqué dans la Table 4.9. La ReLU et la LeakyReLU sont approximées par deux polynômes à coefficients fixes (Table 4.9), tandis que la Softmax est approximée par un polynôme de degré 2 dont les coefficients sont appris au cours du processus d'entraînement du Multi-GAT. Pour mieux orienter l'approximation de la Softmax, elle est initialisée avec les coefficients figurant dans la Table 4.9.

Nous utilisons la méthode des moindres carrés dans l'intervalle $[-6, 6]$ pour générer cinq approximations polynomiales de différents degrés (de 2 à 6) pour les fonctions ReLU et LeakyReLU. Nous remplaçons ensuite les trois fonctions non-linéaires dans le Multi-GAT par ces polynômes générés et par d'autres proposés dans la littérature. Nous évaluons ensuite les modèles obtenus et nous sélectionnons les approximations qui produisent les meilleures performances. Les polynômes choisis sont détaillés dans la Table 4.9. Les approximations de la ReLU et de la LeakyReLU sont également illustrées dans la Figure 4.12.

TABLE 4.9 – Approximations polynomiales des fonctions d'activation.

Fonction d'activation	Approximation polynomiale
ReLU	$0.765 + 0.499x + 0.0574x^2 + 2.2865e^{-11}x^3$
LeakyReLU	$0.382 + 0.6x + 8.048e^{-2}x^2 - 1.101e^{-17}x^3 - 6.056e^{-4}x^4$
Softmax	$0.25 + 0.5x + 0.125x^2$



(a) Comparaison de la fonction ReLU et de son approximation polynomiale.

(b) Comparaison de la fonction LeakyReLU et de son approximation polynomiale.

FIGURE 4.12 – Comparaison des fonctions ReLU et LeakyReLU et de leurs approximations polynomiales.

Dans le calcul matriciel de base du Multi-GAT, les informations de voisinage dans le graphe sont prises en compte en ajoutant un masque (un terme basé sur la matrice d'adjacence A) aux coefficients d'attention avant de les introduire dans la Softmax. Le calcul du masque correspond à $mask = -10 * 10^9(1 - A)$ et l'entrée de la Softmax devient donc : $Softmax_j(e_{ij} + mask)$. Cette formule signifie que l'on ajoute un très grand nombre négatif $(-10 * 10^9)$ à l'exponentielle de la

Softmax, ce qui donne des valeurs qui tendent vers zéro pour tous les nœuds ayant une valeur égale à 0 dans la matrice d'adjacence.

Étant donné que l'approximation polynomiale de la Softmax n'utilise pas l'exponentielle, le calcul matriciel du Multi-GAT doit être mis à jour. Après le remplacement de la Softmax par son approximation polynomiale ($Poly\_Softmax$), la prise en compte du voisinage dans le graphe se fait en multipliant le résultat de l'approximation par la matrice d'adjacence A ($Poly\_Softmax * A$).

Approximation de la couche de normalisation

Nous remplaçons chaque couche de normalisation par son approximation, comme le montre l'Algorithme 4. Nous commençons par mettre à l'échelle l'entrée de la normalisation en la multipliant par x_scale qui est une valeur comprise entre 0 et 1, visant à réduire la plage de variation de l'entrée. Nous calculons ensuite le μ et le σ de la nouvelle entrée. Pour approximer l'inverse carré de $\sigma + \epsilon$ dans la formule de normalisation, nous utilisons la méthode proposée dans [106], illustrée dans la Figure 4.13 et résumée dans l'Algorithme 5.

Algorithme 4 $Poly\_Norm_i$

Entrée : x, x_scale, ϵ
Résultat : y
$x \leftarrow$ x * x_scale
$\mu \leftarrow \frac{1}{N}\sum_{i \in N} x_i$
$\sigma \leftarrow \frac{1}{N}\sum_{i \in N}(x_i - \mu)^2$
$\sigma_{Sqr\_Inv} \leftarrow Sqr\_Inv\_Appr(\sigma + \epsilon)$
$y \leftarrow (x - \mu) * \sigma_{Sqr\_Inv} * \gamma + \beta$
Retourner: y

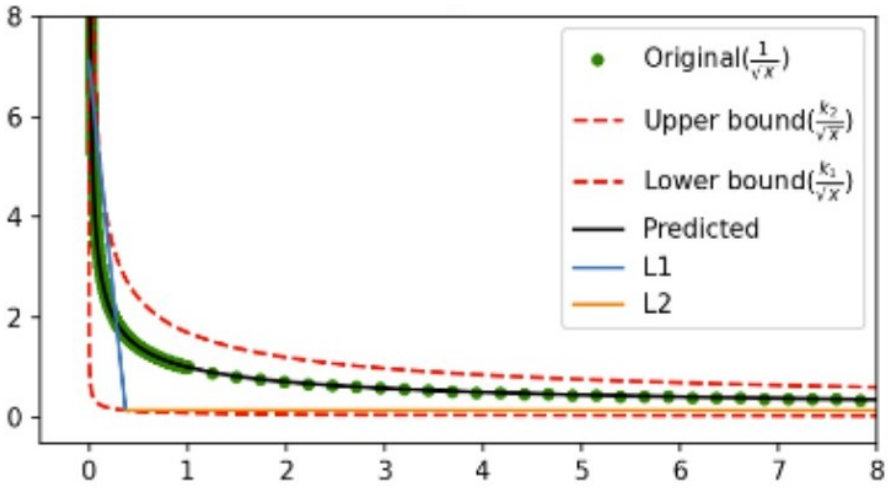


FIGURE 4.13 – Approximation de la fonction inverse proposée par Panda et al. [106].

Ici, nous approximons la fonction sgn par la composition de la fonction proposée dans [21] : $sgn(x) = f_3^{d_f}(x) o g_3^{d_g}(x)$ où : $f_3(x) = \frac{1}{2^4}(35x - 35x^3 + 21x^5 - 5x^7)$ et $g_3(x) = \frac{1}{2^{10}}(4589x - 16577x^3 + 25614x^5 - 12860x^7)$. Nous utilisons les mêmes valeurs de paramètres que celles utilisées dans [106], sur l'intervalle $[a, b] = [10^{-4}, 10^3]$.

Algorithme 5 Sqr_Inv_Appr (Approximation de l'inverse de la racine $\frac{1}{\sqrt{x}}$).

Entrée : $[a, b], \epsilon, d, k_1, k_2, x_1, x_2, P, x, err$
Résultat : $y_d = \frac{1}{\sqrt{x}}$

$\beta(P, x) \leftarrow comp(\frac{P}{b-a}, \frac{x}{b-a})$ ▷ $comp(x, y) = \frac{1+sgn(x-y)}{2}$
$L_1 \leftarrow \frac{-1}{2} k_2 * x_1^{\frac{-3}{2}} * x + \frac{3}{2} \frac{k_2}{\sqrt{x_1}}$
$L_2 \leftarrow \frac{-1}{2} k_2 * x_2^{\frac{-3}{2}} * x + \frac{3}{2} \frac{k_2}{\sqrt{x_2}}$
$h_0(x) \leftarrow (1 + err - \beta(x)) * L_1(x) + (\beta(x) - err) * L_2(x)$
Pour $i \leftarrow 1$ à d **Faire :**
  $y_i \leftarrow \frac{1}{2} * y_{i-1}(x * y_{i-1}^2 + 3$
Fin Pour ▷ Calcul des itérations de Newton pour obtenir y_d
Retourner: y_d

Mode d'apprentissage

Pour obtenir le modèle d'inférence final, nous commençons par pré-entraîner un Multi-GAT partiellement adapté à la HE (nous l'appelons PHE-MG) en remplaçant les trois fonctions d'activation par leurs approximations polynomiales et en conservant les couches de normalisation d'origine. Nous obtenons le premier modèle PHE-MG de base et l'utilisons pour remplacer progressivement les couches de normalisation et obtenir le FHE-MG, comme le montre la Figure 4.14 et comme l'explique l'Algorithme 6.

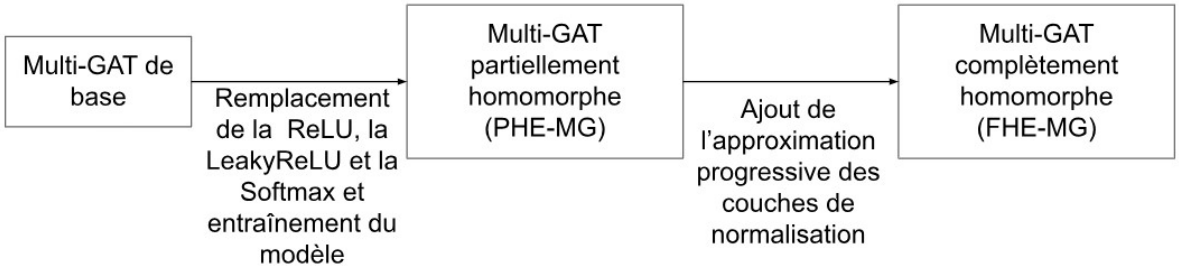


FIGURE 4.14 – Remplacement des opérations d'approximation dans le Multi-GAT de base : nous commençons par remplacer les fonctions d'activation pour former le Multi-GAT partiellement HE, puis en ajoutant les approximations des couches de normalisation, nous obtenons le Multi-GAT entièrement HE final.

À chaque fois que nous remplaçons une couche de normalisation, la distillation des connaissances est utilisée en figeant les poids du modèle à l'exception de la couche d'approximation de la normalisation et en calculant la perte entre la couche de normalisation et sa couche d'approximation. Après chaque remplacement d'une couche de normalisation, nous ajustons les poids du modèle aux nouveaux paramètres de normalisation en affinant toutes les couches du modèle situées entre la couche remplacée et la sortie du modèle, comme le montre la Figure 4.15 (les cases bleues).

Algorithme 6 Le processus d'approximation.

Entrée : MG = PHE-MG, x_scale_tab, ϵ
Résultat : Multi-GAT adapté au HE (FHE-MG)
Pour $i \leftarrow 1$ à nb_norm_layers **Faire :**
Figer les poids de MG
$MG \leftarrow$ MG + $Poly\_Norm_i(x, x\_scale\_tab_i, \epsilon)$
$Loss_{norm} = MSE(Norm_i, Ploy\_Norm_i)$
Optimiser la couche $Poly\_Norm_i$ avec $Loss_{norm}$
$MG \leftarrow$ MG - $Norm_i(x)$
Affiner les couches de MG situées entre $Norm_i$ et la sortie de MG
Fin Pour
Retourner: MG

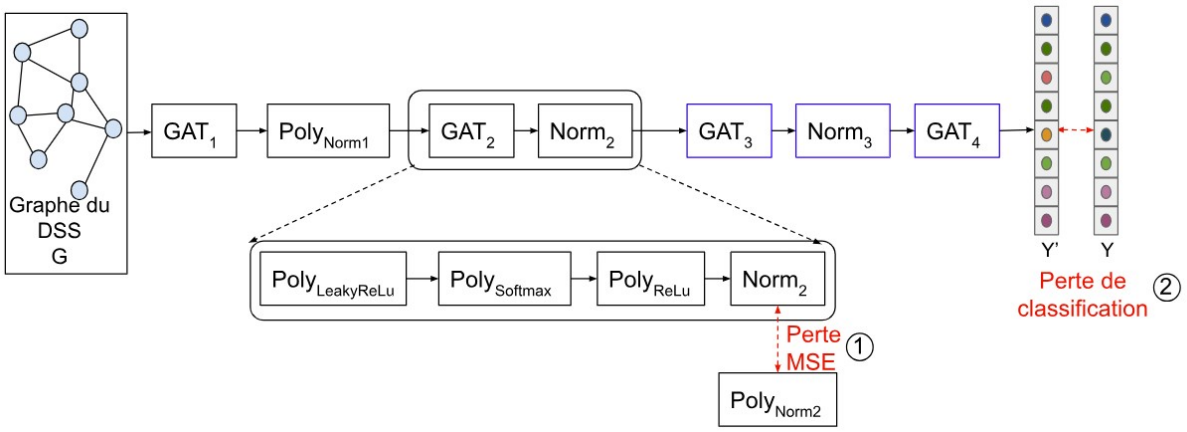


FIGURE 4.15 – Vue d'ensemble du processus d'approximation : l'étape 1 montre l'approximation de normalisation et la distillation de connaissances tandis que l'étape 2 montre l'étape de fine tuning, les boîtes bleues sont les couches à mettre à jour au cours de la deuxième étape.

Modèles finaux retenus

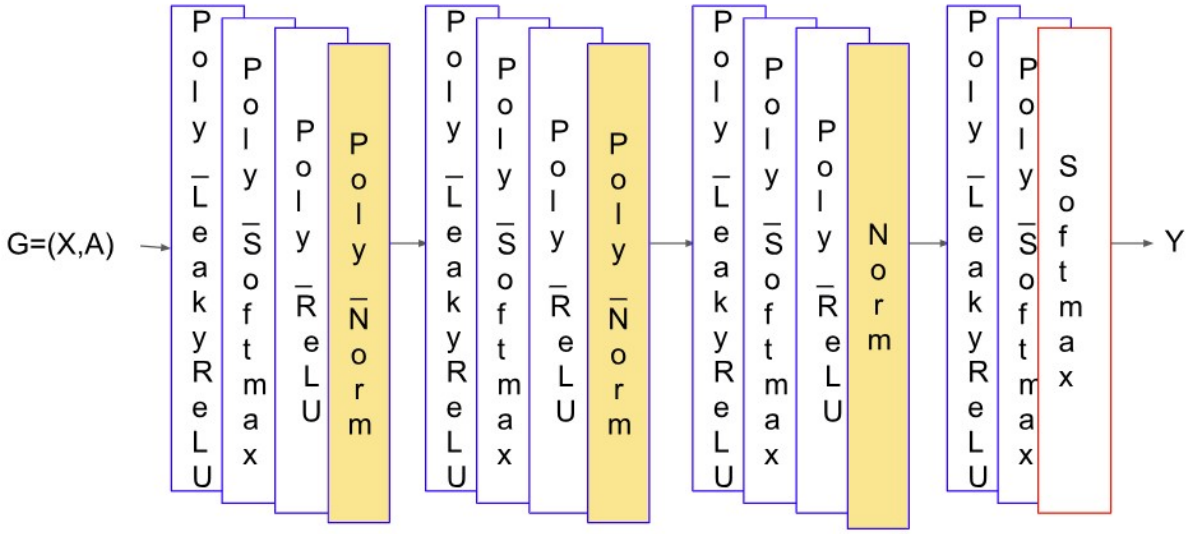
Le modèle final homomorphe (FHE-MG) qu'on obtient après l'approximation des couches de normalisation englobe toutes les approximations polynomiales des fonctions remplacées, à savoir Poly_LeakyReLU, Poly_Softmax, Poly_ReLU, et Poly_Norm.

Ces approximations sont toutes appliquées dans l'ordre illustré dans la Figure 4.16. Nous proposons deux modèles finaux qui peuvent remplacer le Mutli-GAT non homomorphe.

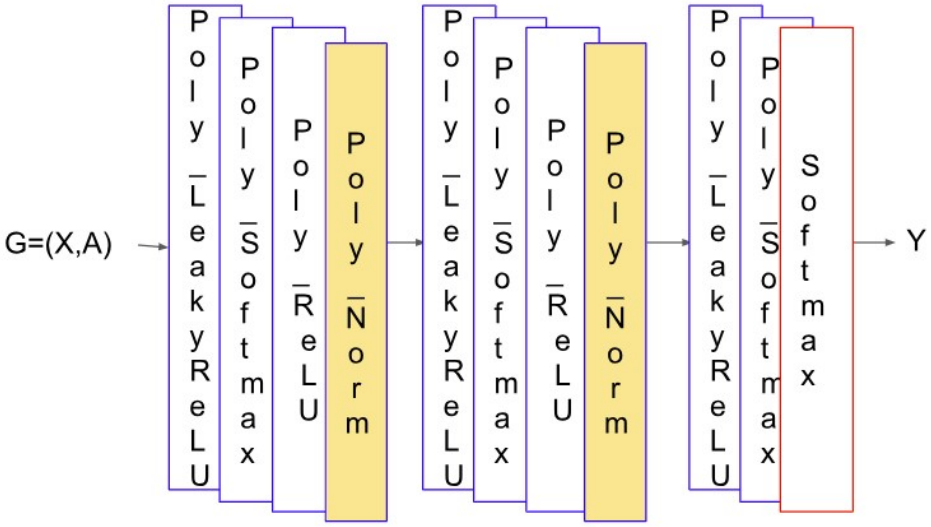
$Multi - GAT_{PHE}$ (Figure 4.16a) : le premier modèle est un Multi-GAT à quatre couches GAT et trois couches de normalisation dont toutes les couches non linaires sont approximées à l'exception de la dernière couche de normalisation qui est maintenue pour optimiser les résultats de la classification et assurer plus de stabilité dans les calculs. Le calcul de cette dernière couche de normalisation peut être ainsi délégué au client.

$Multi - GAT_{FHE}$ (Figure 4.16b) : le deuxième modèle est un Multi-GAT à trois couches GAT et deux couches de normalisation seulement dont toutes les opérations non-linéaires sont approximées. L'objectif de cette réduction du nombre de couches est de réduire le nombre d'opérations d'approximation et par conséquent de limiter les pertes de performances du modèle final.

Le choix entre le modèle $Multi - GAT_{PHE}$ et $Multi - GAT_{FHE}$ se fait selon la priorité choisie. Le modèle $Multi - GAT_{PHE}$ peut garantir des performances meilleures, mais nécessiter



(a) L'ordre des approximations dans le $Multi - GAT_{PHE}$.



(b) L'ordre des approximations dans le $Multi - GAT_{FHE}$.

FIGURE 4.16 – Ordre des approximations polynomiales dans les deux modèles retenus : $Multi - GAT_{PHE}$ et $Multi - GAT_{FHE}$.

une connexion avec le client et un calcul en ligne, ce qui peut être coûteux en temps de calcul et en nécessité de sécuriser le calcul en ligne. $Multi - GAT_{FHE}$ quant à lui, offre un modèle dont tous les calculs sont homomorphes et peuvent se faire sur le serveur, sans nécessité de connexion avec le client. Ce modèle offre néanmoins des performances moins bonnes que celles du premier.

Extension du modèle homomorphe $Multi - GAT_{FHE}$ au mode semi-supervisé

Afin d'améliorer davantage les résultats de l'approximation, nous appliquons l'approche Multi-GAT+VGAE sur le $Multi - GAT_{FHE}$. Nous appliquons l'optimisation semi-supervisée en utilisant le VGAE sur le modèle partiellement homomorphe PHE-MG. Ainsi, nous remplaçons les

fonctions d'activation ReLU, LeakyReLU et Softmax dans le Multi-GAT à trois couches (donnant PHE-MG) et nous l'entraînons pendant « n » étapes. Puis, nous intégrons les composants du VGAE et nous ajoutons les données non annotées, comme le montre la Figure 4.17, et nous poursuivons l'apprentissage.

Le modèle obtenu est alors utilisé comme modèle PHE-MG de base pour remplacer les couches de normalisation suivant le même processus expliqué précédemment.

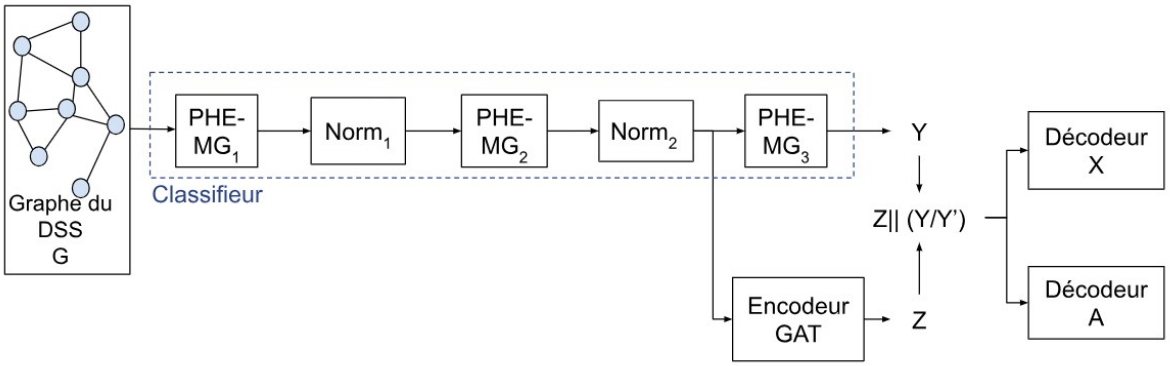


FIGURE 4.17 – Extension du modèle Multi-GAT partiellement homomorphe (PHE-MG) au mode semi-supervisé.

4.3.5 Expérimentations et résultats

Jeux de données utilisés

Dans cette partie, nous évaluons notre approche sur les jeux de données Gen-Invoices-Fr, Gen-Invoices-En, Gen-Payslips et SROIE.

Approximation polynomiale de la ReLU et la LeakyReLU

Dans cette expérimentation, nous faisons varier les degrés d'approximation des polynômes ReLU (voir la Figure 4.18) et LeakyReLU (voir la Figure 4.19) ainsi que la nature de leurs coefficients (fixes ou apprenables), comme le montre la Table 4.10.

Comme le montre la Table 4.10, les meilleures approximations de la ReLU et de la LeakyReLU pour les deux jeux de données (Gen-Invoices-En et SROIE), sont obtenues par les approximations polynomiales à coefficients fixes de degré 4.

Comme le montrent les Figures 4.18 et 4.19, ces polynômes sont pratiquement les plus proches de la ReLU et de la LeakyReLU, en particulier sur l'intervalle $[0, 1]$.

Approximation polynomiale de la Softmax

Nous comparons les différentes approximations Softmax proposées dans la littérature sur notre PHE-MG comme indiqué dans la Table 4.11.

Comme le montre la Table 4.11, l'approximation qui correspond le mieux au jeu de données Gen-Invoices-En est celle proposée dans [31] qui est un polynôme de degré 2 avec des coefficients apprenables. Pour SROIE, le polynôme à coefficients apprenables de degré 5 proposé dans [66], donne les meilleurs résultats, mais le résultat est très proche du polynôme de degré 2 proposé par Dathathri et al. [31] également. Nous conserverons donc ce dernier car il nécessite moins de

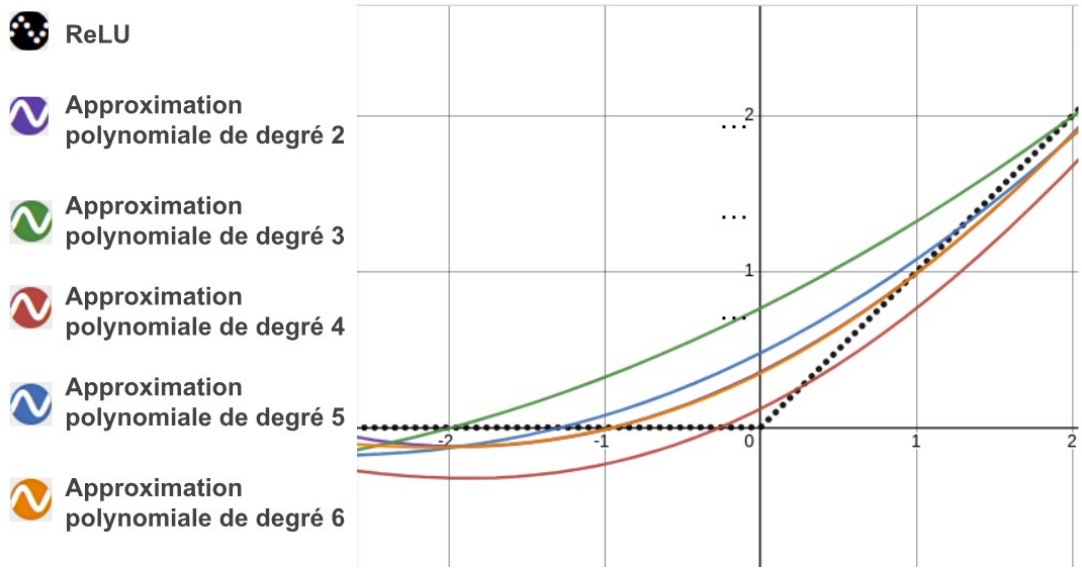


FIGURE 4.18 – Comparaison entre les différents polynômes à coefficients fixes approximant la ReLU.

TABLE 4.10 – Score F1 (%) obtenu sur les deux jeux de données (Gen-Invoices-En et SROIE) en variant les approximations polynomiales ReLU et LeakyReLU (Fx se réfère aux coefficients polynomiaux fixes et Lr aux coefficients appris).

Jeu de données	Fonction	Degré				
		2	3	4	5	6
Gen-Invoices-En	ReLU_Fx	99.15	99.05	**99.31**	99.28	99.04
	ReLU_Lr	99.14	**99.16**	99.14	99.14	99.06
	LeakyReLU_Fx	99.09	98.98	**99.14**	98.78	90.38
	LeakyReLU_Lr	**99.10**	82.51	71.24	67.71	42.08
SROIE	ReLU_Fx	98.21	98.30	**98.36**	98.20	98.21
	ReLU_Lr	**98.37**	98.31	98.13	98.05	97.97
	LeakyReLU_Fx	98.41	98.30	**98.43**	98.42	98.11
	LeakyReLU_Lr	97.70	**97.74**	93.08	92.18	92.93

TABLE 4.11 – Score F1 (%) obtenu en variant l'approximation polynomiale de la Softmax sur les jeux de données Gen-Invoices-En et SROIE.

Approximation de la Softmax	[4]	[3]	[31]	[66]
Gen-Invoices-En	99.20	98.94	**99.33**	98.77
SROIE	98.23	98.30	**98.30**	**98.35**

ressources et de paramètres tout en donnant des résultats aussi bons que le polynôme de degré 5. Pour plus de stabilité pendant le processus d'apprentissage, nous choisissons d'initialiser le polynôme de degré 2 avec les coefficients polynomiaux proposés dans [3].

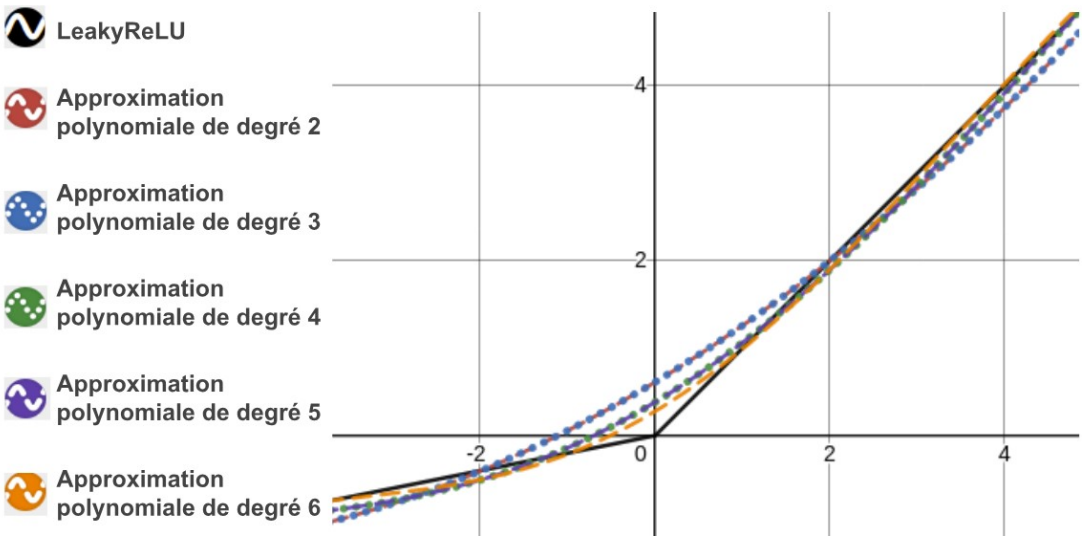


FIGURE 4.19 – Comparaison entre les différents polynômes à coefficients fixes approximant la LeakyReLU.

Effet de la normalisation sur l'approximation

Nous comparons le comportement du modèle avec l'approximation polynomiale des fonctions d'activation en l'absence et en la présence de la normalisation entre les couches GAT. On évalue cela en remplaçant la ReLU dans le Multi-GAT par les différentes approximations polynomiales à coefficients fixes, comme illustré dans la Table 4.12.

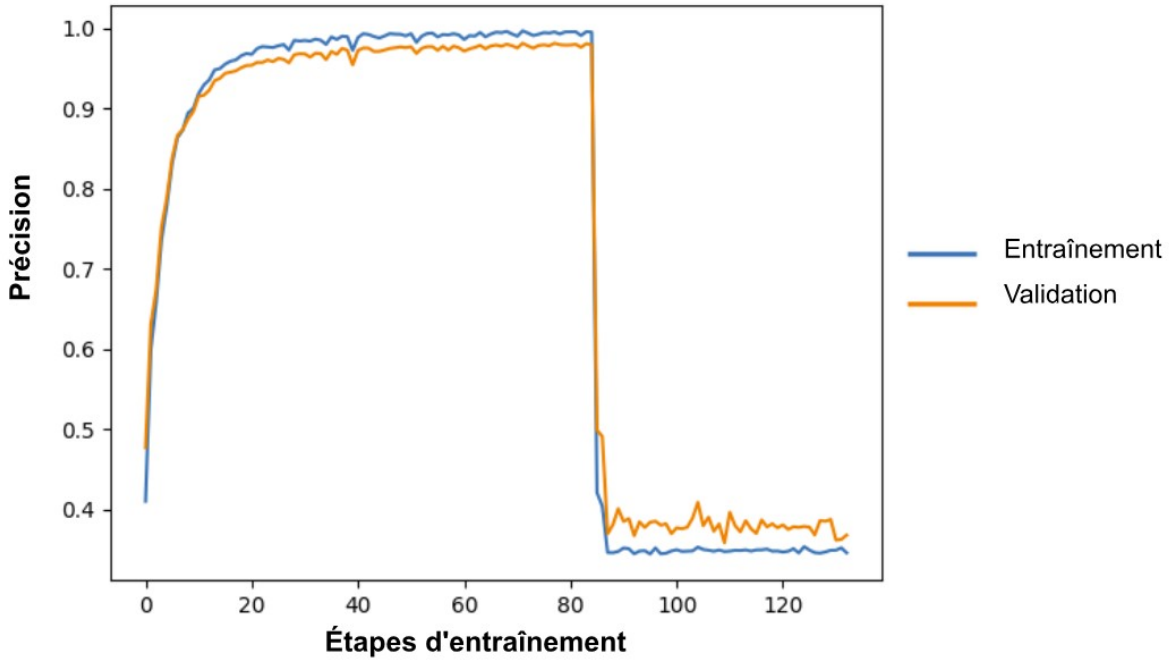


FIGURE 4.20 – Phénomène de l'explosion du gradient lors de l'entraînement du Multi-GAT avec un polynôme de degré 6.

Fonction d'activation	Normalisation	ReLU	Polynôme de degré				
			2	3	4	5	6
Gen-Invoices-En	✗	**98.41**	98.32	98.05	97.19	97.77	97.66
	✓	99.00	99.15	99.05	**99.31**	99.28	99.04
SROIE	✗	**98.22**	98.22	98.10	98.08	97.90	95.82
	✓	98.40	98.12	98.30	**98.36**	98.20	98.21

TABLE 4.12 – Score F1 (%) obtenu en variant le degré des approximations polynomiales de la ReLU en présence et en l'absence de la normalisation.

Comme on peut le voir dans la Table 4.12, l'ajout de la normalisation améliore les résultats du Multi-GAT et permet de mieux stabiliser l'approximation de la ReLU dans le modèle. De plus, les approximations polynomiales de degré supérieur à 4 présentent des chutes brutales de performances lors de l'entraînement sans normalisation comme on peut le voir dans la Figure 4.20, produites à cause du phénomène de l'explosion du gradient.

Comme on peut l'observer, à une étape n de l'entraînement, les performances chutent et les poids du modèle neuronal passent à « nul ». En effet, cela apparaît lorsque les valeurs des gradients deviennent importantes, par conséquent leur multiplication devient énorme avec le temps et cela produit un modèle incapable d'apprendre avec un comportement instable. Une des solutions pour ce problème, est d'ajouter des couches de normalisation dans le Multi-GAT après les approximations dans l'objectif de normaliser et de remettre à l'échelle les valeurs dans le Multi-GAT et éviter l'explosion des résultats de multiplication. Comme on peut le remarquer dans la Figure 4.21 après l'ajout de la couche de normalisation, le comportement du modèle devient plus stable et il n'y a pas de chute de performances.

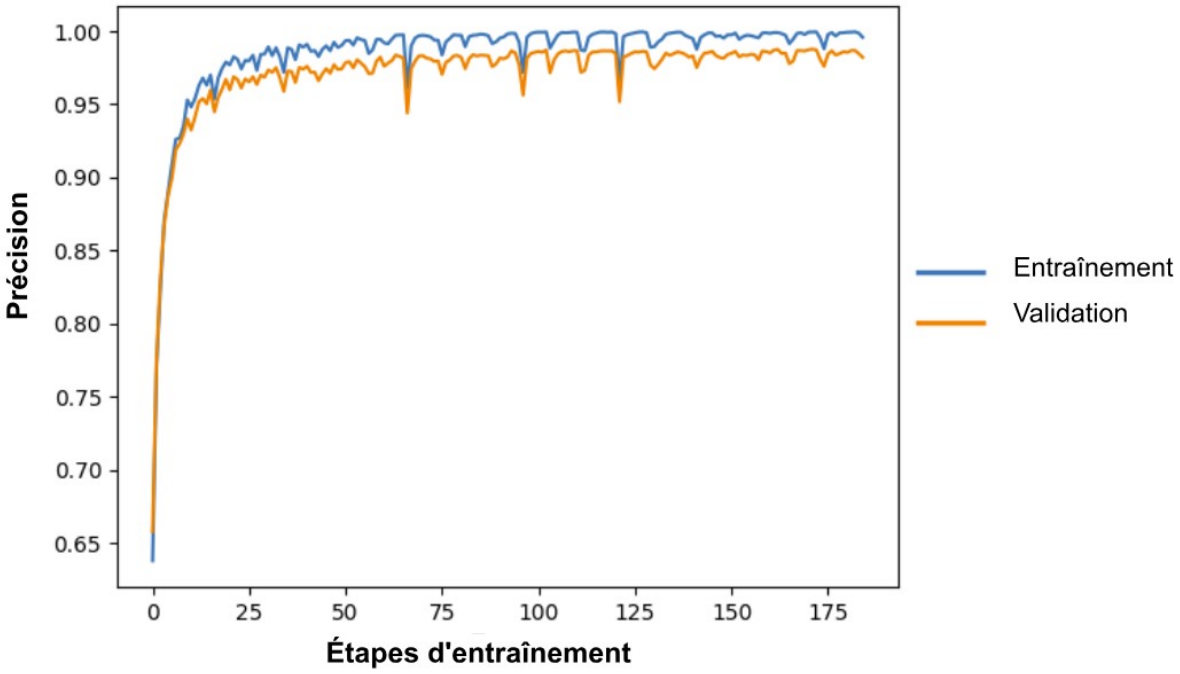


FIGURE 4.21 – Comportement stable du Multi-GAT après ajout de la normalisation.

Approximation de la couche de normalisation

Nous évaluons l'effet de chaque couche de normalisation remplacée avec le raffinement supplémentaire du modèle décrit dans l'Algorithme 4. Nous mettons d'abord en évidence l'effet de chaque remplacement d'une couche de normalisation, comme le montre la Table 4.13.

TABLE 4.13 – Score F1 (%) obtenu sur les trois jeux de données en remplaçant progressivement les trois couches de normalisation.

Jeu de données	PHE-MG	Numéro de la couche de normalisation		
		1	1 et 2	1, 2 et 3
Gen-Invoices-En	99.25	99.35	98.97	68.16
Gen-Invoices-Fr	98.14	98.14	98.10	63.04
SROIE	98.42	89.78	89.58	70.32

Les résultats de la Table 4.13 montrent que le remplacement des deux premières couches de normalisation ne fait pas perdre beaucoup de performances au modèle (près de 1 % pour les jeux de données Invoices et 9 % pour SROIE), mais le remplacement des trois couches ensemble (donnant le $Multi - GAT_{FHE}$) dégrade considérablement les performances.

Pour garantir un modèle entièrement homomorphe sans perte de performance significative, nous présentons deux stratégies distinctes. La première stratégie consiste à déléguer la dernière couche de normalisation au client qui la calcule et envoie les résultats au serveur. La seconde stratégie consiste à réduire le nombre de couches GAT à trois, avec deux couches de normalisation.

La deuxième stratégie donnant le modèle $Multi - GAT_{FHE}$ fournit les résultats présentés dans la Table 4.14.

TABLE 4.14 – Score F1 (%) sur les trois jeux de données en remplaçant progressivement les trois couches de normalisation dans un PHE-MG à 3 couches. Le résultat est un $Multi - GAT_{FHE}$.

Jeu de données	PHE-MG	Numéros des couches approximées	
		1	1 et 2
Gen-Invoices-En	97.55	97.25	95.96
Gen-Invoices-Fr	98.57	98.40	95.89
Gen-Payslips	99.85	98.20	95.16
SROIE	98.30	94.35	93.41

Comme le montre la Table 4.14, la perte de performance est moindre si l'on réduit le nombre de couches dans le PHE-MG (environ 3 % de perte pour les factures et les fiches de paie et 5 % pour SROIE). Ainsi, en fonction des priorités du modèle, c'est-à-dire effectuer tous les calculs sur le serveur ou conserver le maximum de performances, nous pouvons choisir entre les deux stratégies.

Nous avons également comparé notre méthode d'approximation de la normalisation (Poly_Norm), expliquée dans l'Algorithme 4, avec celle proposée par Chen et al. dans [17]. Ainsi, nous avons comparé le score F1 (%) du PHE-MG à trois couches GAT après le remplacement de la première couche de normalisation.

Comme on peut le constater dans la Table 4.15, sur les trois jeux de données Gen-Invoices-En, Gen-Invoices-Fr et Gen-Payslips, notre approximation de la normalisation est plus performante que celle proposée dans [17]. Cette dernière adopte une formule linéaire trop simple qui ne prend pas en compte la moyenne ou la variance des données. Quant à notre proposition, elle maintient

Jeu de données	Poly_Norm	$x * \gamma + \beta$ [17]
Gen-Invoices-En	93.20	86.11
Gen-Invoices-Fr	93.99	87.82
Gen-Payslips	93.78	80.89

TABLE 4.15 – Comparaison du score F1 (%) obtenu sur un Multi-GAT à trois couches GAT après l'approximation de la première couche de normalisation par la méthode Poly_Norm et l'approximation proposée dans [17].

l'utilisation de la moyenne et la variance dans la formule d'approximation, et elle reproduit approximativement le comportement de la fonction de normalisation de base.

Ordre de remplacement des couches de normalisation

Nous avons également évalué l'ordre de remplacement des couches de normalisation dans le PHE-MG. Nous avons en effet analysé le remplacement par ordre décroissant des couches du PHE-MG à quatre couches GAT et à trois couches.

TABLE 4.16 – Score F1 (%) obtenu avec un PHE-MG à quatre couches GAT en remplaçant les couches de normalisation par ordre décroissant.

Jeu de données	PHE-MG	Numéro de la couche de normalisation		
		3	2 et 3	1, 2 et 3
Invoices-En	99.46	98.95	92.71	68.92
SROIE	98.42	98.30	96.68	86.52

TABLE 4.17 – Score F1 (%) obtenu avec un PHE-MG à trois couches GAT en remplaçant les couches de normalisation par ordre décroissant.

Jeu de données	PHE-MG	Numéro de la couche de normalisation	
		2	1 et 2
Gen-Invoices-En	97.55	93.28	55.81
SROIE	98.01	96.18	89.51

Comme on peut le voir dans les Tables 4.16 et 4.17, le remplacement par ordre décroissant est moins efficace que celui de l'ordre croissant présenté dans les expérimentations précédentes. La différence est très apparente pour le PHE-MG à trois couches GAT contenant deux couches de normalisation. Si on fige les paramètres des dernières couches de normalisation, elles ne peuvent pas s'adapter à leurs nouvelles entrées approximées. Cela explique la perte considérable observée après le remplacement de toutes les couches de normalisation, contrairement au remplacement des couches par ordre croissant qui adapte les couches de sortie aux nouvelles couches d'approximation de la normalisation.

Effet du fine tuning sur les résultats de l'approximation de la normalisation

Nous évaluons ici l'effet, sur les performances du modèle, de l'ajout de l'étape de fine tuning de la partie du PHE-MG, située entre la couche de normalisation et la sortie du PHE-MG, après remplacement de chaque couche de normalisation.

Jeu de données	Fine tuning après Norm 1		Fine tuning après Norm 2	
	✗	✓	✗	✓
Gen-Invoices-En	93.20	97.25	74.19	95.96
Gen-Invoices-Fr	93.99	98.40	63.38	95.89
Gen-Payslips	93.78	98.20	82.36	95.16
SROIE	94.13	94.35	79.17	93.41

TABLE 4.18 – Score F1 (%) obtenu sur l'approximation de la couche de normalisation dans un PHE-MG à trois couches GAT.

Comme on peut le voir dans la Table 4.18, l'étape de fine tuning qu'on ajoute après le remplacement de chaque couche de normalisation et qu'on applique sur la partie qui suit la couche remplacée, améliore grandement les résultats. Cela permet aux couches de sortie de s'adapter à la nouvelle normalisation apprise et à mieux effectuer la tâche de classification.

Mode semi-supervisé

Dans cette expérimentation, nous montrons l'effet de l'utilisation de l'architecture semi-supervisée présentée dans le Chapitre 3 sur le Multi-GAT homomorphe.

Jeu de données	Mode supervisé		Mode semi-supervisé	
	PHE-MG	FHE-MG	PHE-MG	FHE-MG
Gen-Invoices-En	97.55	95.96	97.88	**96.43**
Gen-Invoices-Fr	98.57	95.89	98.65	**95.98**

TABLE 4.19 – Comparaison entre les scores F1 (%) obtenus avec et sans apprentissage semi-supervisé

Nous intégrons le VGAE avec le modèle PHE-MG que nous entraînons en mode semi-supervisé, ensuite nous appliquons l'algorithme d'approximation de la normalisation sur le modèle résultant, fournissant ainsi le FHE-MG final.

Les résultats présentés dans la Table 4.19 démontrent que l'ajout de l'apprentissage semi-supervisé améliore les résultats du modèle FHE-MG final. En effet, avec l'apprentissage semi-supervisé, nous augmentons les performances du modèle initial (PHE-MG) avant de procéder à l'approximation des couches de normalisation. Ainsi, même si nous perdons encore en performance en appliquant l'approximation de la normalisation, nous maximisons les résultats du modèle de base sur lequel nous appliquons l'approximation de la normalisation, ce qui permet d'améliorer le résultat du modèle final.

Résultats globaux sur SROIE

Nous comparons les résultats et la complexité du modèle FHE-MG avec différents systèmes d'extraction d'informations de de la littérature sur le jeu de données SROIE.

Comme le montre la Table 4.20, le modèle FHE-MG que nous proposons permet d'obtenir assez de bonnes performances par rapport aux autres systèmes. Il surpasse la version Base de BERT. FHE-MG est aussi beaucoup moins complexe que les autres modèles. Il ne nécessite pas une étape de pré-entraînement, comme les autres modèles basés sur des transformers. Contrairement aux autres modèles, le notre est le seul adapté au chiffrement homomorphe. Les

Système	Paramètres	Score F1
BERT(B) [36]	340M	92.00
LayoutLM (B) [135]	113M	94.38
LayoutLM (L) [135]	343M	95.24
LayoutLMv2 (B) [134]	200M	96.25
StrucText [84]	107M	96.88
LayoutLMv2(L) [134]	426M	97.81
LAMBERT [41]	125M	98.17
FHE-MG (notre modèle)	42M	93.41

TABLE 4.20 – Comparaison du score F1 (%) et de la complexité entre le modèle FHE-MG et d'autres systèmes de pointe. M désigne un million, B le modèle de base et L le modèle large.

autres modèles utilisent des transformers avec le mécanisme d'attention qui comporte plusieurs opérations non-linéaires et non adaptées au chiffrement homomorphe.

Résultats globaux sur les jeux de données de factures

Nous présentons dans cette section les résultats détaillés de l'approximation polynomiale sur Gen-Invoices-Fr et Gen-Invoices-En. Nous comparons également les performances obtenues par le PHE-MG (ayant trois couches GAT et deux couches de normalisation) avec celles de FHE-MG.

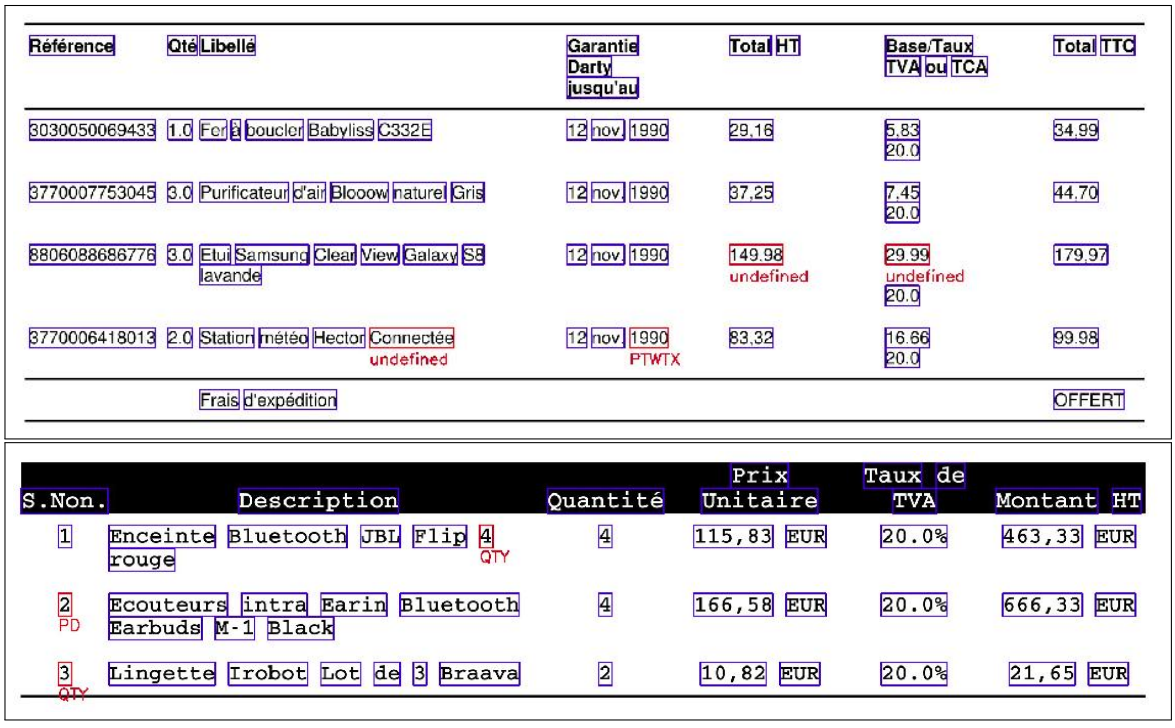


FIGURE 4.22 – Deux exemples d'erreurs engendrées par le FHE-MG sur les factures (Gen-Invoices-Fr). Les mots encadrés en rouge correspondent aux erreurs de classification.

La Table 4.21 présente les performances du modèle PHE-MG sur le jeu de données Gen-Invoices-Fr, qui obtient des résultats intéressants avec un score F1 d'au moins 94%. Les pertes observées pour ce modèle sont dues à la confusion entre certaines classes d'entités qui partagent

TABLE 4.21 – Score F1 (%) obtenu sur le jeu de données Gen-Invoices-Fr, en comparant le PHE-MG et FHE-MG.

Entité	PHE-MG	FHE-MG
Invoice date	98.69	94.51
Invoice number	98.82	95.00
Order number	96.40	91.67
Payment mode	99.80	95.86
Company name	94.65	88.80
Company address	97.53	92.44
Company SIREN number	98.21	89.56
Company SIRET number	99.57	91.57
Company VAT number	98.60	93.13
Company APE code	97.95	88.63
Company Contact number	98.23	90.14
Company fax number	98.13	91.47
Client number	96.44	94.97
Client billing name	97.90	95.56
Client billing address	94.84	92.85
Client shipping name	97.74	96.60
Client shipping address	94.15	93.42
Product serial number	97.57	94.94
Product description	99.08	97.88
Product unit price	98.79	96.03
Product quantity	96.92	92.87
Product price without tax	98.36	94.59
Tax rate	98.24	91.99
Total without tax	99.40	96.87
Total tax amount	99.18	97.27
Net payable amount	99.78	98.01

des similitudes en termes de contenu, de position et de mots-clés introductifs. La sous-section suivante présente une description détaillée de ces confusions.

L'utilisation du modèle FHE-MG augmente en effet la confusion entre les différentes classes de mots, ce qui réduit les performances du modèle. Lors de l'approximation de la normalisation, les valeurs des vecteurs de caractéristiques des mots à la sortie de chaque couche perdent de leur précision, ce qui accentue encore les confusions.

Les deux exemples de la Figure 4.22 montrent des mots qui ont été bien classés par le PHE-MG, mais qui sont mal classés par le FHE-MG. Dans le premier exemple, les mots « connectée », « 149,98 » et « 29,99 » qui appartiennent respectivement aux classes « Product description » (PD), « Total without tax » (TWTX), et « Tax rate » (TXR) sont classés comme « undefined ». En revanche, le mot « 1990 » qui appartient à la classe « undefined » est classé comme « Product price without tax » (PTWTX). Le deuxième exemple illustre les erreurs résultant de la confusion entre les deux classes « Product description » (PD) et « Product Quantity » (QTY).

On remarque également des scores importants sur l'ensemble Gen-Invoices-En (voir la Table 4.22). Comme pour Gen-Invoices-Fr, des entités qui présentent des similarités dans le contenu, les mots-clés introductifs et les positions dans le DSS peuvent provoquer des confusions lors de

TABLE 4.22 – Score F1 (%) obtenu sur le jeu de données Gen-Invoices-En en comparant le PHE-MG et FHE-MG.

Entité	PHE-MG	FHE-MG
Invoice date	99.49	96.92
Invoice number	99.42	95.35
Order number	99.67	93.73
Payment mode	99.57	98.40
Company name	96.12	91.87
Company address	98.26	96.52
Company SIREN number	95.77	92.64
Company SIRET number	97.78	92.80
Company VAT number	98.41	96.47
Company APE code	97.82	98.11
Company Contact number	98.20	96.70
Company fax number	97.40	96.01
Client number	99.37	89.52
Client billing name	97.42	94.84
Client billing address	89.46	88.63
Client shipping name	98.11	97.37
Client shipping address	88.21	87.98
Product serial number	91.07	84.58
Product description	99.46	98.41
Product unit price	99.35	98.46
Product quantity	95.21	89.82
Product price without tax	99.00	97.31
Tax rate	99.95	98.85
Total without tax	99.90	99.04
Total tax amount	99.90	99.90
Net payable amount	100	99.62

la classification des mots de l'ensemble Gen-Invoices-En.

La Figure 4.23 montre des exemples d'erreurs produites par l'utilisation de FHE-MG sur Gen-Invoices-Fr. Ces erreurs n'étaient pas générées par l'utilisation de PHE-MG.

Dans le premier exemple de la Figure 4.23, le modèle PHE-MG a classé correctement les mots « Cannes » et « 704-928 » qui appartiennent respectivement aux classes « Client billing address » (BA) et « Order number » (ONUM). Cependant, en raison de la diminution de la précision du modèle après l'approximation, ces mots se retrouvent mal classés et confondus avec d'autres classes en utilisant le modèle FHE-MG. « Cannes » qui appartient à la classe « BA » est confondu avec la classe « Client shipping address » (SHA). Nous constatons que le contenu de ces deux entités est exactement le même et leurs emplacements dans le DSS sont proches, ce qui entraîne leur confusion. Le mot « 704-928 » qui appartient initialement à l'entité « ONUM » est classé comme « undefined ». Dans le deuxième exemple de la Figure 4.23, les deux mots « INV-2632 » et « COM-292 » appartenant principalement aux classes « Invoice number » (IN) et « ONUM », sont également mal classés par le FHE-MG comme « undefined » et « Invoice date » (IDATE).

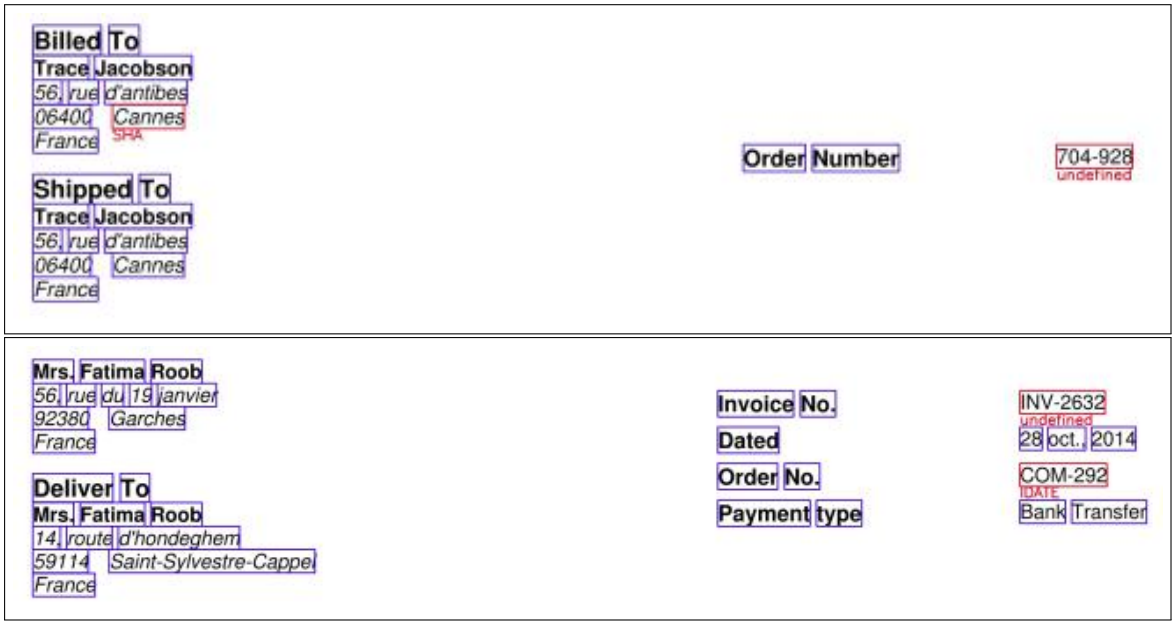


FIGURE 4.23 – Deux exemples d'erreurs engendrées par le FHE-MG sur les factures (Gen-Invoices-En). Les mots encadrés en bleu correspondent aux mots bien classés. Les mots encadrés en rouge représentent les erreurs de classification et les étiquettes affichées en-dessous des mots représentent les prédictions.

Analyse des erreurs

Dans cette section, nous analysons les différentes erreurs engendrées sur quelques DSSs du Gen-Invoices-En et Gen-Invoices-Fr par les systèmes Multi-GAT de base ayant trois couches GAT et deux couches de normalisation, le PHE-MG et le FHE-MG. La Figure 4.24 présente des exemples d'erreurs de classification résultant du modèle Multi-GAT de base et qui s'accentuent en appliquant l'approximation avec les deux modèles PHE-MG et FHE-MG.

Les pertes enregistrées sur le modèle de base sont dues à la confusion entre certaines classes d'entités dont le contenu et les mots-clés introductifs sont semblables, les positions dans le DSS sont également très similaires et ils partagent le même voisinage également. Nous recensons dans ce qui suit quelques confusions observées dans les exemples de la Figure 4.24.

— Confusion entre les entités « Order number » (ONUM), « Client number » (CNUM) et « Invoice number » (IN). Elles partagent généralement le même format et se retrouvent dans le même voisinage (exemples 1 et 2 dans la Figure 4.24).

— D'autres entités comme « Invoice number » (IN) peuvent être faussement classées comme « undefined » (voir les exemples 3 et 4). Cela vient de leur confusion avec d'autres codes ayant le même format et représentant essentiellement des informations facultatives classées principalement comme « undefined ».

— L'entité « Client shipping address » (SHA) est confondue avec « Client billing address » (BA) (exemple 5 et 6). Ces deux informations ont généralement le même contenu et se positionnent aux mêmes endroits.

— Une confusion entre l'entité « Company Contact number » (SFAX) et « Company fax number » (SCN) qui ont habituellement la même valeur et se trouvent dans des positions très proches (exemple 7).

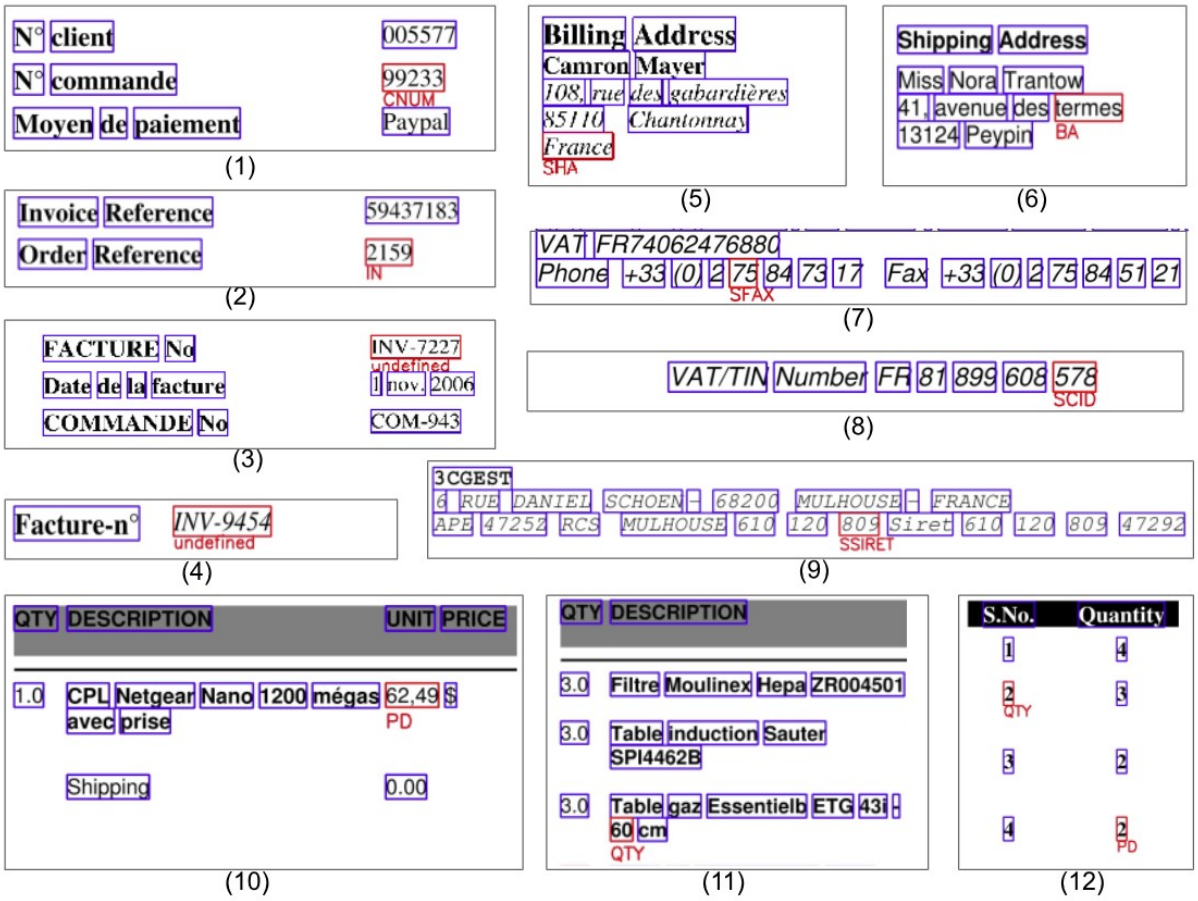


FIGURE 4.24 – Exemples d'erreurs engendrées par le Multi-GAT de base ayant trois couches GAT et deux couches de normalisation. Les mots encadrés en bleu correspondent aux mots bien classés. Les mots encadrés en rouge représentent les erreurs de classification et les étiquettes affichées en-dessous des mots représentent leurs prédictions.

— Une confusion peut être repérée également entre la classe « Company Siret number » (SSIRET), « Company VAT number » (SVAT) et « Comanpy identification number » (SCID) (exemples 8 et 9).

— Les entités contenues dans le tableau des produits peuvent également générer des confusions. « Product description » (PD), « Product quantity » (QTY) et « Product unit price » (PU) se positionnent l'une à coté de l'autre et ont des formats similaires (des champs numériques). Les exemples 10, 11 et 12 de la Figure 4.24 illustrent ces confusions.

4.3.6 Synthèse

Dans cette section, nous avons présenté un modèle entièrement homomorphe pour l'extraction d'informations à partir de DSSs. Nous avons proposé des approximations polynomiales de faible degré pour les trois fonctions d'activation dans le Multi-GAT, la ReLU, la LeakyReLU et la Softmax, du mécanisme d'attention, sans perte de performances. Nous avons également proposé une approximation de la couche de normalisation qui ne repose pas sur des données d'entraînement et qui peut être entièrement calculée sur le serveur, évitant ainsi toute connexion avec le

détenteur des données. Notre modèle propose des approximations de faible degré pour toutes les opérations non homomorphes, ce qui permet d'éviter les connexions multiples avec le détenteur des données. Notre système classe avec précision 27 entités à partir de deux jeux de données de factures ainsi qu'un jeu de données de fiches de paie avec un score F1 global intéressant de 95 %, et enregistre un score remarquable de 93,41 % sur l'ensemble des données SROIE.

4.4 Conclusion

Afin de préparer le modèle d'inférence au chiffrement homomorphe, nous avons proposé dans ce chapitre une approche visant à réduire la dimensionnalité des données et la complexité du modèle, ainsi qu'une méthode permettant d'approximer les opérations non homomorphes dans le Multi-GAT.

Nous avons ainsi présenté une méthode simple qui utilise la moyenne des vecteurs de plongements textuels des sous-mots afin de fusionner les caractéristiques textuelles et de réduire la complexité du modèle. En outre, nous avons introduit un réseau basé sur des couches denses qui fusionne les différentes modalités du texte, y compris le plongement du texte, la position et les caractéristiques visuelles.

De plus, nous avons adapté notre modèle d'inférence au chiffrement homomorphe. Pour ce faire, nous avons utilisé des approximations polynomiales pour les fonctions non-linéaires dans le Multi-GAT, notamment ReLU, LeakyReLU et Softmax, ainsi qu'une couche de normalisation que nous avons ajoutée au modèle de base. Différentes techniques, telles que le fine-tuning et la distillation des connaissances, ont été utilisées pour améliorer les performances des approximations. L'approche proposée minimise les pertes de classification et ne nécessite aucun calcul à déléguer au client. Elle y parvient grâce à l'utilisation d'approximations polynomiales de faible degré.

5

Conclusion et perspectives

Dans cette thèse, nous avons étudié des approches peu complexes, répondant au problème d'EI à partir de documents administratifs semi-structurés. L'objectif principal de ce travail était de proposer un système pour l'EI dans les DSSs satisfaisant plusieurs contraintes, notamment le manque de jeux de données d'entraînement publics annotés, la possibilité d'exploiter des données non annotées du domaine, la mise en place d'un système peu complexe par rapport aux systèmes de l'état de l'art, généralement basés sur des transfomers qui nécessitent des centaines de millions de paramètres et une vaste base de données de pré-entraînement. Nous avons également étudié une approche de préparation du système d'inférence pour le chiffrement homomorphe qui assure la confidentialité des données de l'utilisateur.

Étant donné qu'il n'existe pas de jeux de données d'entraînement public pour ce type de document en raison de leur confidentialité, nous avons tout d'abord proposé un générateur générique de documents semi-structurés avec plusieurs métriques pour évaluer la variation du contenu et de la mise en page. Nous avons ensuite mis en œuvre trois cas d'utilisation, à savoir les tickets de caisse, les factures et les fiches de paie.

Le premier modèle d'EI que nous avons proposé est le $Multi - GAT_{Dataset}$ qui modélise le document sous forme de graphe est basé sur le mécanisme d'attention multi-têtes. Nous avons montré qu'il existe une corrélation entre le nombre de classes à extraire et le nombre de têtes d'attention dans le réseau d'attention à graphes. Une deuxième version inductive multimodale $Multi - GAT_{DSS}$ a aussi été présentée, intégrant d'autres modalités telles que la région et les caractéristiques visuelles. Dans cette version, nous avons limité la taille des matrices modélisant les graphes, tout en veillant à ce que le voisinage immédiat des mots soit toujours pris en compte. Ce processus permet de traiter tous les documents quel que soit le nombre de mots qu'ils contiennent, réduisant ainsi le temps de traitement des documents et donc les ressources de calcul requises.

Ce modèle a été étendu au mode semi-supervisé par l'ajout d'un composant neuronal génératif (VGAE). Cela a permis d'améliorer les performances du classifieur en apprenant simultanément sur des données annotées et non annotées du même domaine. La fonction de perte du VGAE a également été paramétrée en fonction de la perte de classification, afin de mieux orienter le modèle global vers l'optimisation de la tâche de classification.

En vue de réduire davantage la complexité du modèle, nous avons proposé une méthode simple pour fusionner les caractéristiques textuelles des sous-mots composant les mots entiers, ainsi qu'un réseau simplifié qui fusionne les différentes modalités du texte tout en éliminant les caractéristiques redondantes. Enfin, nous avons élaboré une approche permettant d'adapter notre modèle d'inférence au chiffrement homomorphe. Pour ce faire, nous avons proposé des

approximations polynomiales pour les différentes fonctions non-linéaires utilisées dans le Multi-GAT. L'approche proposée minimise les pertes de classification et ne nécessite pas de connexion avec le client tout en utilisant des approximations polynomiales de faible degré.

Afin d'atteindre l'objectif final du projet dans lequel cette thèse est inscrite, et qui vise à effectuer une analyse complète des DSSs, plusieurs points peuvent être envisagés dans le futur. Certains d'entre eux concernent l'intégration des travaux réalisés dans les autres parties du projet, tandis que d'autres concernent l'amélioration et la validation des approches proposées.

Une première perspective qui concerne la génération des données synthétiques peut inclure l'étude et la génération d'autres types de DSS, comme les relevés bancaires. Cela permettra d'étendre les différentes approches proposées et de les valider sur davantage de types de DSSs.

La deuxième perspective est de mettre les modèles et structures proposés au service de la tâche de classification des documents, qui est la tâche qui précède l'extraction d'informations. Cela revient à exploiter la modélisation des DSSs que nous avons proposée ainsi que le mécanisme d'attention pour prédire les classes des DSSs avant de les transmettre au Multi-GAT en vue de l'application de l'EI.

La troisième perspective est de tester les différentes approches proposées sur plus de données numérisées réelles afin de prouver leur efficacité. Cela permettra, par exemple, de tester les modèles sur des jeux de données susceptibles de présenter des erreurs d'OCR. Cette tâche peut être réalisée après un premier déploiement du Multi-GAT et une collection des données des utilisateurs avec leurs consentement. Ces documents peuvent ensuite être utilisés pour tester et affiner les modèles entraînés.

Enfin, il serait intéressant de réaliser des tests supplémentaires pour réduire davantage le nombre de paramètres dans les approximations polynomiales des opérations non linéaires dans le Multi-GAT, ce qui permettrait de le rendre encore plus léger. En outre, les performances de notre système peuvent être validées en appliquant le protocole du chiffrement homomorphe aux données et au modèle final proposé (FHE-MG).

A

Liste de publications

— Belhadj, D., Belaïd, Y., & Belaïd, A. (2021, September). Automatic generation of semi-structured documents. In Document Analysis and Recognition–ICDAR 2021 Workshops, Proceedings, Part II 16 (pp. 191-205). Springer International Publishing.

— Belhadj, D., Belaïd, Y., & Belaïd, A. (2021, September). Consideration of the Word's Neighborhood in GATs for Information Extraction in Semi-structured Documents. In Document Analysis and Recognition–ICDAR 2021, Proceedings, Part II 16 (pp. 854-869). Springer International Publishing.

— Belhadj, D., Belaïd, A., & Belaïd, Y. (2023, August). Improving Information Extraction from Semi-structured Documents Using Attention Based Semi-variational Graph Auto-Encoder. In International Conference on Document Analysis and Recognition–ICDAR 2023 (pp. 113-129). Cham : Springer Nature Switzerland.

— Belhadj, D., Belaïd, A., & Belaïd, Y. (2023, September). Low-Dimensionality Information Extraction Model for Semi-structured Documents. In International Conference on Computer Analysis of Images and Patterns–CAIP 2023 (pp. 76-85). Cham : Springer Nature Switzerland.

— Belhadj, D., Belaïd, A., & Belaïd, Y. (2024, February). Homomorphic encryption friendly Multi-GAT for information extraction in business documents. In The International Conference on Pattern Recognition Applications and Methods–ICPRAM 2024.

Bibliographie

[1] Abd-Alsabour, N. : On the role of dimensionality reduction. J. Comput. **13**(5), 571–579 (2018)

[2] Abdi, H., Williams, L.J. : Principal component analysis. WIREs Computational Statistics **2**(4), 433–459 (2010). https ://doi.org/https ://doi.org/10.1002/wics.101

[3] Al Badawi, A., Hoang, L., Mun, C.F., Laine, K., Aung, K.M.M. : Privft : Private and fast text classification with homomorphic encryption. IEEE Access **8**, 226544–226556 (2020)

[4] Ali, H., Javed, R.T., Qayyum, A., AlGhadhban, A., Alazmi, M., Alzamil, A., Al-utaibi, K., Qadir, J. : Spam-das : Secure and privacy-aware misinformation detection as a service. TechRxiv (2022)

[5] Ali, R.E., So, J., Avestimehr, A.S. : On polynomial approximations for privacy-preserving and verifiable relu networks. arXiv preprint arXiv :2011.05530 (2020)

[6] Balazs, J.A., Velásquez, J.D. : Opinion mining and information fusion : a survey. Information Fusion **27**, 95–110 (2016)

[7] Baruch, M., Drucker, N., Greenberg, L., Moshkowich, G. : A methodology for training homomorphic encryption friendly neural networks. In : International Conference on Applied Cryptography and Network Security. pp. 536–553. Springer (2022)

[8] Belhadj, D., Belaïd, Y., Belaïd, A. : Consideration of the word's neighborhood in gats for information extraction in semi-structured documents. In : Document Analysis and Recognition–ICDAR 2021 : 16th International Conference, Lausanne, Switzerland, September 5–10, 2021, Proceedings, Part II 16. pp. 854–869. Springer (2021)

[9] Bird, S., Klein, E., Loper, E. : Natural language processing with Python : analyzing text with the natural language toolkit. " O'Reilly Media, Inc." (2009)

[10] Biswas, S., Riba, P., Lladós, J., Pal, U. : Docsynth : a layout guided approach for controllable document image synthesis. In : International Conference on Document Analysis and Recognition. pp. 555–568. Springer (2021)

[11] Blanchard, J., Belaïd, Y., Belaïd, A. : Automatic generation of a custom corpora for invoice analysis and recognition. In : 2019 International Conference on Document Analysis and Recognition Workshops (ICDARW). vol. 7, pp. 1–1. IEEE (2019)

[12] Bojanowski, P., Grave, E., Joulin, A., Mikolov, T. : Enriching word vectors with subword information. Transactions of the association for computational linguistics **5**, 135–146 (2017)

[13] Brakerski, Z., Gentry, C., Vaikuntanathan, V. : Fully homomorphic encryption without bootstrapping cryptology eprint archive, paper 2011/277 (2011)

[14] Burkov, E., Lempitsky, V. : Deep neural networks with box convolutions. Advances in Neural Information Processing Systems **31** (2018)

[15] Capobianco, S., Marinai, S. : Docemul : a toolkit to generate structured historical documents. In : 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR). vol. 1, pp. 1186–1191. IEEE (2017)

[16] Chen, M., Tang, Q., Livescu, K., Gimpel, K. : Variational sequential labelers for semi-supervised learning. In : Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. pp. 215—-226 (2019)

[17] Chen, T., Bao, H., Huang, S., Dong, L., Jiao, B., Jiang, D., Zhou, H., Li, J., Wei, F. : Thex : Privacy-preserving transformer inference with homomorphic encryption. In : Findings of the Association for Computational Linguistics : ACL 2022. pp. 3510–3520 (2022)

[18] Cheng, X., Xu, W., Wang, T., Chu, W. : Variational semi-supervised aspect-term sentiment analysis via transformer. In : Proceedings of the 23rd Conference on Computational Natural Language Learning. pp. 961—-969 (2019)

[19] Cheng, Z., Zhang, P., Li, C., Liang, Q., Xu, Y., Li, P., Pu, S., Niu, Y., Wu, F. : Trie++ : Towards end-to-end information extraction from visually rich documents. arXiv preprint arXiv :2207.06744 (2022)

[20] Cheon, J.H., Kim, A., Kim, M., Song, Y. : Homomorphic encryption for arithmetic of approximate numbers. In : Advances in Cryptology–ASIACRYPT 2017 : 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I 23. pp. 409–437. Springer (2017)

[21] Cheon, J.H., Kim, D., Kim, D. : Efficient homomorphic comparison methods with optimal complexity. In : Advances in Cryptology–ASIACRYPT 2020 : 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7–11, 2020, Proceedings, Part II 26. pp. 221–256. Springer (2020)

[22] Chiang, J. : On polynomial approximation of activation function. CoRR (2022)

[23] Chiang, J. : Privacy-preserving cnn training with transfer learning. CoRR (2023)

[24] Chillotti, I., Gama, N., Georgieva, M., Izabachène, M. : Tfhe : fast fully homomorphic encryption over the torus. Journal of Cryptology **33**(1), 34–91 (2020)

[25] Chiu, J.P., Nichols, E. : Named entity recognition with bidirectional lstm-cnns. Transactions of the association for computational linguistics **4**, 357–370 (2016)

[26] Chong, Y., Ding, Y., Yan, Q., Pan, S. : Graph-based semi-supervised learning : A review. Neurocomputing **408**, 216–230 (2020)

[27] Chung, J., Gulcehre, C., Cho, K., Bengio, Y. : Empirical evaluation of gated recurrent neural networks on sequence modeling. In : NIPS 2014 Workshop on Deep Learning, December 2014 (2014)

[28] Dang, T.A.N., Nguyen, D.T. : End-to-end information extraction by character-level embedding and multi-stage attentional u-net. In : Proceedings of the 30th British Machine Vision Conference 2019, BMVC. p. 96 (2019)

[29] Dash, M., Liu, H., Yao, J. : Dimensionality reduction of unsupervised data. In : Proceedings ninth ieee international conference on tools with artificial intelligence. pp. 532–539. IEEE (1997)

[30] Dash, M., Liu, H. : Feature selection for classification. Intelligent data analysis **1**(1-4), 131–156 (1997)

[31] Dathathri, R., Saarikivi, O., Chen, H., Laine, K., Lauter, K., Maleki, S., Musuvathi, M., Mytkowicz, T. : Chet : an optimizing compiler for fully-homomorphic neural-network inferencing. In : Proceedings of the 40th ACM SIGPLAN conference on programming language design and implementation. pp. 142–156 (2019)

[32] Dauphin, Y.N., Fan, A., Auli, M., Grangier, D. : Language modeling with gated convolutional networks. In : International conference on machine learning. pp. 933–941. PMLR (2017)

[33] Davis, P.J. : Interpolation and approximation. Courier Corporation (1975)

[34] Deng, G., Duan, X., Tang, M., Zhang, Y., Huang, Y. : Non-interactive and privacy-preserving neural network learning using functional encryption. Future Generation Computer Systems **145**, 454–465 (2023)

[35] Denk, T.I., Reisswig, C. : Bertgrid : Contextualized embedding for 2d document representation and understanding. In : Workshop on Document Intelligence at NeurIPS 2019 (2019)

[36] Devlin, J., Chang, M.W., Lee, K., Toutanova, K. : Bert : Pre-training of deep bidirectional transformers for language understanding. In : Proceedings of North American Chapter of the Association for Computational Linguistics : Human Language Technologies. (2019)

[37] Dos Santos, C., Gatti, M. : Deep convolutional neural networks for sentiment analysis of short texts. In : Proceedings of COLING 2014, the 25th international conference on computational linguistics : technical papers. pp. 69–78 (2014)

[38] Duda, R., Hart, P., Stork, D., Ionescu, A. : Pattern classification, chapter nonparametric techniques (2000)

[39] Fan, J., Vercauteren, F. : Somewhat practical fully homomorphic encryption. Cryptology ePrint Archive (2012)

[40] Felhi, G., Le Roux, J., Seddah, D. : Challenging the semi-supervised vae framework for text classification. In : Proceedings of the Second Workshop on Insights from Negative Results in NLP. pp. 136–143 (2021)

[41] Garncarek, Ł., Powalski, R., Stanisławek, T., Topolski, B., Halama, P., Turski, M., Graliński, F. : Lambert : layout-aware language modeling for information extraction. In : International Conference on Document Analysis and Recognition. pp. 532–547. Springer (2021)

[42] Gertrudes, J.C., Zimek, A., Sander, J., Campello, R.J. : A unified framework of density-based clustering for semi-supervised classification. In : Proceedings of the 30th international conference on scientific and statistical database management. pp. 1–12 (2018)

[43] Ghodsi, Z., Gu, T., Garg, S. : Safetynets : Verifiable execution of deep neural networks on an untrusted cloud. Advances in Neural Information Processing Systems **30** (2017)

[44] Giachanou, A., Gonzalo, J., Mele, I., Crestani, F. : Sentiment propagation for predicting reputation polarity. In : Advances in Information Retrieval : 39th European Conference on IR Research, ECIR 2017, Aberdeen, UK, April 8-13, 2017, Proceedings 39. pp. 226–238. Springer (2017)

[45] Gilad-Bachrach, R., Dowlin, N., Laine, K., Lauter, K., Naehrig, M., Wernsing, J. : Cryptonets : Applying neural networks to encrypted data with high throughput and accuracy. In : International conference on machine learning. pp. 201–210. PMLR (2016)

[46] Gilmer, J., Schoenholz, S.S., Riley, P.F., Vinyals, O., Dahl, G.E. : Neural message passing for quantum chemistry. In : International conference on machine learning. pp. 1263–1272. PMLR (2017)

[47] Grattarola, D., Alippi, C. : Graph neural networks in tensorflow and keras with spektral. IEEE Computational Intelligence Magazine **16**(1), 99–106 (2021)

[48] Gupta, K., Lazarow, J., Achille, A., Davis, L.S., Mahadevan, V., Shrivastava, A. : Layouttransformer : Layout generation and completion with self-attention. In : Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 1004–1014 (2021)

[49] Gururangan, S., Dang, T., Card, D., Smith, N.A. : Variational pretraining for semi-supervised text classification. In : Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. pp. 5880—-5894 (2019)

[50] Hamilton, W., Ying, Z., Leskovec, J. : Inductive representation learning on large graphs. Advances in neural information processing systems **30** (2017)

[51] Hammami, M., Héroux, P., Adam, S., d'Andecy, V.P. : One-shot field spotting on colored forms using subgraph isomorphism. In : 2015 13th International Conference on Document Analysis and Recognition (ICDAR). pp. 586–590. IEEE (2015)

[52] He, K., Gkioxari, G., Dollár, P., Girshick, R. : Mask r-cnn. In : Proceedings of the IEEE international conference on computer vision. pp. 2961–2969 (2017)

[53] He, K., Zhang, X., Ren, S., Sun, J. : Deep residual learning for image recognition. In : Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)

[54] Heinzerling, B., Strube, M. : Bpemb : Tokenization-free pre-trained subword embeddings in 275 languages. In : Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018) (2018)

[55] Hira, Z.M., Gillies, D.F. : A review of feature selection and feature extraction methods applied on microarray data. Advances in bioinformatics **2015** (2015)

[56] Hochreiter, S., Schmidhuber, J. : Long short-term memory. Neural computation **9**(8), 1735–1780 (1997)

[57] Hua, Y., Huang, Z., Guo, J., Qiu, W. : Attention-based graph neural network with global context awareness for document understanding. In : China National Conference on Chinese Computational Linguistics. pp. 45–56. Springer (2020)

[58] Huang, L., Ma, D., Li, S., Zhang, X., Wang, H. : Text level graph neural network for text classification. 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). pp. 3444–3450 (2019)

[59] Huang, X., Wu, L., Ye, Y. : A review on dimensionality reduction techniques. International Journal of Pattern Recognition and Artificial Intelligence **33**(10), 1950017 (2019)

[60] Huang, Z., Chen, K., He, J., Bai, X., Karatzas, D., Lu, S., Jawahar, C. : Icdar2019 competition on scanned receipt ocr and information extraction. In : 2019 International Conference on Document Analysis and Recognition (ICDAR). pp. 1516–1520. IEEE (2019)

[61] Huang, Z., Xu, W., Yu, K. : Bidirectional lstm-crf models for sequence tagging. arXiv preprint arXiv :1508.01991 (2015)

[62] Ibarrondo, A., Önen, M. : Fhe-compatible batch normalization for privacy preserving deep learning. In : Data Privacy Management, Cryptocurrencies and Blockchain Technology :

ESORICS 2018 International Workshops, DPM 2018 and CBT 2018, Barcelona, Spain, September 6-7, 2018, Proceedings 13. pp. 389–404. Springer (2018)

[63] Ioffe, S., Szegedy, C. : Batch normalization : Accelerating deep network training by reducing internal covariate shift. In : International conference on machine learning. pp. 448–456. pmlr (2015)

[64] Ishiyama, T., Suzuki, T., Yamana, H. : Highly accurate cnn inference using approximate activation functions over homomorphic encryption. In : 2020 IEEE International Conference on Big Data (Big Data). pp. 3989–3995. IEEE (2020)

[65] Jain, D., Singh, V. : Feature selection and classification systems for chronic disease prediction : A review. Egyptian Informatics Journal **19**(3), 179–189 (2018)

[66] Jang, J., Lee, Y., Kim, A., Na, B., Yhee, D., Lee, B., Cheon, J.H., Yoon, S. : Privacy-preserving deep sequential model with matrix homomorphic encryption. In : Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security. pp. 377–391 (2022)

[67] Jean, N., Xie, S.M., Ermon, S. : Semi-supervised deep kernel learning : Regression with unlabeled data by minimizing predictive variance. Advances in Neural Information Processing Systems **31** (2018)

[68] Jia, W., Sun, M., Lian, J., Hou, S. : Feature dimensionality reduction : a review. Complex & Intelligent Systems **8**(3), 2663–2693 (2022)

[69] Jindal, P., Kumar, D. : A review on dimensionality reduction techniques. Int. J. Comput. Appl **173**(2), 42–46 (2017)

[70] Kadhim, A.I. : An evaluation of preprocessing techniques for text classification. International Journal of Computer Science and Information Security (IJCSIS) **16**(6), 22–32 (2018)

[71] Katti, A.R., Reisswig, C., Guder, C., Brarda, S., Bickel, S., Höhne, J., Faddoul, J.B. : Chargrid : Towards understanding 2d documents. In : Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 4459–4469 (2018)

[72] Kerroumi, M., Sayem, O., Shabou, A. : Visualwordgrid : Information extraction from scanned documents using a multimodal approach. In : International Conference on Document Analysis and Recognition. pp. 389–402. Springer (2021)

[73] Khaire, U.M., Dhanalakshmi, R. : Stability of feature selection algorithm : A review. Journal of King Saud University-Computer and Information Sciences **34**(4), 1060–1073 (2022)

[74] Kim, A., Song, Y., Kim, M., Lee, K., Cheon, J.H. : Logistic regression model training based on the approximate homomorphic encryption. BMC medical genomics **11**(4), 23–31 (2018)

[75] Kingma, D.P., Mohamed, S., Jimenez Rezende, D., Welling, M. : Semi-supervised learning with deep generative models. Advances in neural information processing systems **27** (2014)

[76] Kipf, T.N., Welling, M. : Semi-supervised classification with graph convolutional networks. In : Proceedings of the 5th International Conference on Learning Representations, (ICLR) (2016)

[77] Kipf, T.N., Welling, M. : Variational graph auto-encoders. CoRR (2016)

[78] Lafferty, J., McCallum, A., Pereira, F.C. : Conditional random fields : Probabilistic models for segmenting and labeling sequence data. In : Proceedings of the Eighteenth International Conference on Machine Learning (ICML). pp. 282–289 (2001)

[79] Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., Dyer, C. : Neural architectures for named entity recognition. In : North American Chapter of the Association for Computational Linguistics (2016)

[80] Langley, P., et al. : Selection of relevant features in machine learning. In : Proceedings of the AAAI Fall symposium on relevance. vol. 184, pp. 245–271. California (1994)

[81] Le, A.D., Van Pham, D., Nguyen, T.A. : Deep learning approach for receipt recognition. In : International Conference on Future Data and Security Engineering. pp. 705–712. Springer (2019)

[82] Li, D., Wang, H., Shao, R., Guo, H., Xing, E., Zhang, H. : Mpcformer : fast, performant and private transformer inference with mpc. In : The Eleventh International Conference on Learning Representations (2022)

[83] Li, Y., Zemel, R., Brockschmidt, M., Tarlow, D. : Gated graph sequence neural networks. In : Proceedings of the 4th International Conference on Learning Representations, ICLR 2016 (2016)

[84] Li, Y., Qian, Y., Yu, Y., Qin, X., Zhang, C., Liu, Y., Yao, K., Han, J., Liu, J., Ding, E. : Structext : Structured text understanding with multi-modal transformers. In : Proceedings of the 29th ACM International Conference on Multimedia. pp. 1912–1920 (2021)

[85] Liao, R., Brockschmidt, M., Tarlow, D., Gaunt, A.L., Urtasun, R., Zemel, R. : Graph partition neural networks for semi-supervised classification. In : 6th International Conference on Learning Representations, ICLR 2018 (2018)

[86] Liao, Z., Luo, J., Gao, W., Zhang, Y., Zhang, W. : Homomorphic cnn for privacy preserving learning on encrypted sensor data. In : 2019 Chinese Automation Congress (CAC). pp. 5593–5598. IEEE (2019)

[87] Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S. : Feature pyramid networks for object detection. In : Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2117–2125 (2017)

[88] Lin, W., Gao, Q., Sun, L., Zhong, Z., Hu, K., Ren, Q., Huo, Q. : Vibertgrid : a jointly trained multi-modal 2d document representation for key information extraction from documents. In : International Conference on Document Analysis and Recognition. pp. 548–563. Springer (2021)

[89] Lin, Y., Meng, Y., Sun, X., Han, Q., Kuang, K., Li, J., Wu, F. : Bertgcn : Transductive text classification by combining gnn and bert. In : Findings of the Association for Computational Linguistics : ACL-IJCNLP 2021. pp. 1456–1462 (2021)

[90] Ling, M., Chen, J., Möller, T., Isenberg, P., Isenberg, T., Sedlmair, M., Laramee, R.S., Shen, H.W., Wu, J., Giles, C.L. : Document domain randomization for deep learning document layout extraction. In : Document Analysis and Recognition–ICDAR 2021 : 16th International Conference, Lausanne, Switzerland, September 5–10, 2021, Proceedings, Part I 16. pp. 497–513. Springer (2021)

[91] Liu, J., Juuti, M., Lu, Y., Asokan, N. : Oblivious neural network predictions via minionn transformations. In : Proceedings of the 2017 ACM SIGSAC conference on computer and communications security. pp. 619–631 (2017)

[92] Liu, X., Gao, F., Zhang, Q., Zhao, H. : Graph convolution for multimodal information extraction from visually rich documents. In : Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, Volume 2 (Industry Papers). pp. 32–39 (2019)

[93] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V. : Roberta : A robustly optimized bert pretraining approach. CoRR (2019)

[94] Lohani, D., Belaïd, A., Belaïd, Y. : An invoice reading system using a graph convolutional network. In : Computer Vision–ACCV 2018 Workshops : 14th Asian Conference on Computer Vision, Perth, Australia, December 2–6, 2018, Revised Selected Papers 14. pp. 144–158. Springer (2019)

[95] Lou, Q., Jiang, L. : Hemet : A homomorphic-encryption-friendly privacy-preserving mobile neural network architecture. In : International conference on machine learning. pp. 7102–7110. PMLR (2021)

[96] Ma, X., Hovy, E. : End-to-end sequence labeling via bi-directional lstm-cnns-crf. In : Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL (2016)

[97] Ma, X., Hovy, E. : End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In : Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers). pp. 1064–1074 (2016)

[98] Manning, C.D., Surdeanu, M., Bauer, J., Finkel, J.R., Bethard, S., McClosky, D. : The stanford corenlp natural language processing toolkit. In : Proceedings of 52nd annual meeting of the association for computational linguistics : system demonstrations. pp. 55–60 (2014)

[99] Meinardus, G. : Approximation of functions : Theory and numerical methods, vol. 13. Springer Science & Business Media (2012)

[100] Mejova, Y., Srinivasan, P. : Exploring feature definition and selection for sentiment classifiers. In : Proceedings of the International AAAI Conference on Web and Social Media. pp. 546–549 (2011)

[101] Mikolov, T., Chen, K., Corrado, G.S., Dean, J. : Efficient estimation of word representations in vector space. In : 1st International Conference on Learning Representations, ICLR (2013)

[102] Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J. : Distributed representations of words and phrases and their compositionality. Advances in neural information processing systems **26** (2013)

[103] Mohassel, P., Zhang, Y. : Secureml : A system for scalable privacy-preserving machine learning. In : 2017 IEEE symposium on security and privacy (SP). pp. 19–38. IEEE (2017)

[104] Moore, B. : Principal component analysis in linear systems : Controllability, observability, and model reduction. IEEE transactions on automatic control **26**(1), 17–32 (1981)

[105] Naseem, U., Razzak, I., Khan, S.K., Prasad, M. : A comprehensive survey on word representation models : From classical to state-of-the-art word representation language models. Transactions on Asian and Low-Resource Language Information Processing **20**(5), 1–35 (2021)

[106] Panda, S. : Polynomial approximation of inverse sqrt function for fhe. In : International Symposium on Cyber Security, Cryptology, and Machine Learning. pp. 366–376. Springer (2022)

[107] Papineni, K., Roukos, S., Ward, T., Zhu, W.J. : Bleu : a method for automatic evaluation of machine translation. In : Proceedings of the 40th annual meeting of the Association for Computational Linguistics. pp. 311–318 (2002)

[108] Patil, A.G., Ben-Eliezer, O., Perel, O., Averbuch-Elor, H. : Read : Recursive autoencoders for document layout generation. In : Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops. pp. 544–545 (2020)

[109] Pechenizkiy, M., Tsymbal, A., Puuronen, S. : On combining principal components with fisher's linear discriminants for supervised learning. Foundations of Computing and Decision Sciences **31**(1), 59–74 (2006)

[110] Peng, H., Li, J., He, Y., Liu, Y., Bao, M., Wang, L., Song, Y., Yang, Q. : Large-scale hierarchical text classification with recursively regularized deep graph-cnn. In : Proceedings of the 2018 world wide web conference. pp. 1063–1072 (2018)

[111] Peng, Y., Wu, Z., Jiang, J. : A novel feature selection approach for biomedical data classification. Journal of Biomedical Informatics **43**(1), 15–23 (2010)

[112] Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L. : Deep contextualized word representations. In : Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, Volume 1 (Long Papers). pp. 2227–2237 (2018)

[113] Pisaneschi, L., Gemelli, A., Marinai, S. : Automatic generation of scientific papers for data augmentation in document layout analysis. Pattern Recognition Letters **167**, 38–44 (2023)

[114] Qian, J., Zhang, P., Zhu, H., Liu, M., Wang, J., Ma, X. : Lhdnn : Maintaining high precision and low latency inference of deep neural networks on encrypted data. Applied Sciences **13**(8), 4815 (2023)

[115] Qian, Y., Santus, E., Jin, Z., Guo, J., Barzilay, R. : Graphie : A graph-based framework for information extraction. In : Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, Volume 1 (Long and Short Papers). pp. 751–761 (2019)

[116] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. : Language models are unsupervised multitask learners. OpenAI blog **1**(8), 9 (2019)

[117] Raman, N., Shah, S., Veloso, M. : Synthetic document generator for annotation-free layout recognition. Pattern Recognition **128**, 108660 (2022)

[118] Ren, S., He, K., Girshick, R., Sun, J. : Faster r-cnn : Towards real-time object detection with region proposal networks. Advances in neural information processing systems **28** (2015)

[119] Ronneberger, O., Fischer, P., Brox, T. : U-net : Convolutional networks for biomedical image segmentation. In : International Conference on Medical image computing and computer-assisted intervention. pp. 234–241. Springer (2015)

[120] Rusinol, M., Benkhelfallah, T., Poulain dAndecy, V. : Field extraction from administrative documents by incremental structural templates. In : 2013 12th International Conference on Document Analysis and Recognition. pp. 1100–1104. IEEE (2013)

[121] Sánchez, J., Salgado, A., García, A., Monzón, N. : Idsem, an invoices database of the spanish electricity market. Scientific data **9**(1), 786 (2022)

[122] Sennrich, R., Haddow, B., Birch, A. : Neural machine translation of rare words with subword units. In : 54th Annual Meeting of the Association for Computational Linguistics. pp. 1715–1725. Association for Computational Linguistics (ACL) (2016)

[123] Singh, K.N., Devi, S.D., Devi, H.M., Mahanta, A.K. : A novel approach for dimension reduction using word embedding : An enhanced text classification approach. International Journal of Information Management Data Insights **2**(1), 100061 (2022)

[124] Van Beusekom, J., Keysers, D., Shafait, F., Breuel, T.M. : Distance measures for layout-based document image retrieval. In : Second International Conference on Document Image Analysis for Libraries (DIAL'06). pp. 11–pp. IEEE (2006)

[125] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I. : Attention is all you need. Advances in neural information processing systems **30** (2017)

[126] Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y. : Graph Attention Networks. International Conference on Learning Representations (2018)

[127] Visalakshi, S., Radha, V. : A literature review of feature selection techniques and applications : Review of feature selection in data mining. In : 2014 IEEE international conference on computational intelligence and computing research. pp. 1–6. IEEE (2014)

[128] Wang, C.C., Tu, C.H., Kao, M.C., Hung, S.H. : Tensorhe : a homomorphic encryption transformer for privacy-preserving deep learning. In : Proceedings of the Conference on Research in Adaptive and Convergent Systems. pp. 124–130 (2022)

[129] Wang, J., Liu, C., Jin, L., Tang, G., Zhang, J., Zhang, S., Wang, Q., Wu, Y., Cai, M. : Towards robust visual information extraction in real world : new dataset and novel solution. In : Proceedings of the AAAI Conference on Artificial Intelligence. vol. 35, pp. 2738–2745 (2021)

[130] Wang, K., Han, S.C., Poon, J. : Induct-gcn : Inductive graph convolutional networks for text classification. In : 2022 26th International Conference on Pattern Recognition (ICPR). pp. 1243–1249. IEEE (2022)

[131] Wei, M., He, Y., Zhang, Q. : Robust layout-aware ie for visually rich documents with pre-trained language models. In : Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 2367–2376 (2020)

[132] Wu, F., Souza, A., Zhang, T., Fifty, C., Yu, T., Weinberger, K. : Simplifying graph convolutional networks. In : International conference on machine learning. pp. 6861–6871. PMLR (2019)

[133] Xie, S., Girshick, R., Dollár, P., Tu, Z., He, K. : Aggregated residual transformations for deep neural networks. In : Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1492–1500 (2017)

[134] Xu, Y., Xu, Y., Lv, T., Cui, L., Wei, F., Wang, G., Lu, Y., Florencio, D., Zhang, C., Che, W., et al. : Layoutlmv2 : Multi-modal pre-training for visually-rich document understanding. In : Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1 : Long Papers). pp. 2579–2591 (2021)

[135] Xu, Y., Li, M., Cui, L., Huang, S., Wei, F., Zhou, M. : Layoutlm : Pre-training of text and layout for document image understanding. In : Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 1192–1200 (2020)

[136] Yang, J., Frangi, A.F., Yang, J.y., Zhang, D., Jin, Z. : Kpca plus lda : a complete kernel fisher discriminant framework for feature extraction and recognition. IEEE Transactions on pattern analysis and machine intelligence **27**(2), 230–244 (2005)

[137] Yao, L., Mao, C., Luo, Y. : Graph convolutional networks for text classification. In : Proceedings of the AAAI conference on artificial intelligence. pp. 7370–7377 (2019)

[138] Ye, H., Zhang, W., Nie, M. : An improved semi-supervised variational autoencoder with gate mechanism for text classification. International Journal of Pattern Recognition and Artificial Intelligence (2022)

[139] Yu, W., Lu, N., Qi, X., Gong, P., Xiao, R. : Pick : processing key information extraction from documents using improved graph learning-convolutional networks. In : 2020 25th International Conference on Pattern Recognition (ICPR). pp. 4363–4370. IEEE (2021)

[140] Yu, W., Lu, N., Qi, X., Gong, P., Xiao, R. : Pick : processing key information extraction from documents using improved graph learning-convolutional networks. In : 2020 25th International Conference on Pattern Recognition (ICPR). pp. 4363–4370. IEEE (2021)

[141] Zebari, R., Abdulazeez, A., Zeebaree, D., Zebari, D., Saeed, J. : A comprehensive review of dimensionality reduction techniques for feature selection and feature extraction. Journal of Applied Science and Technology Trends **1**(2), 56–70 (2020)

[142] Zhang, C., Liang, B., Huang, Z., En, M., Han, J., Ding, E., Ding, X. : Look more than once : An accurate detector for text of arbitrary shapes. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 10544–10553 (2019)

[143] Zhang, H., Goodfellow, I., Metaxas, D., Odena, A. : Self-attention generative adversarial networks. In : International conference on machine learning. pp. 7354–7363. PMLR (2019)

[144] Zhang, H., Zhang, J. : Text graph transformer for document classification. In : Conference on empirical methods in natural language processing (EMNLP) (2020)

[145] Zhang, P., Xu, Y., Cheng, Z., Pu, S., Lu, J., Qiao, L., Niu, Y., Wu, F. : Trie : end-to-end text reading and information extraction for document understanding. In : Proceedings of the 28th ACM International Conference on Multimedia. pp. 1413–1422 (2020)

[146] Zhang, Y., Lu, Z. : Exploring semi-supervised variational autoencoders for biomedical relation extraction. Methods **166**, 112–119 (2019)

[147] Zhang, Y., Yu, X., Cui, Z., Wu, S., Wen, Z., Wang, L. : Every document owns its structure : Inductive text classification via graph neural networks. In : Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. pp. 334–339 (2020)

[148] Zhao, N., Mio, W., Liu, X. : A hybrid pca-lda model for dimension reduction. In : The 2011 International Joint Conference on Neural Networks. pp. 2184–2190. IEEE (2011)

[149] Zhao, X., Niu, E., Wu, Z., Wang, X. : Cutie : Learning to understand documents with convolutional universal text information extractor. CoRR (2019)

[150] Zheng, P., Cai, Z., Zeng, H., Huang, J. : Keyword spotting in the homomorphic encrypted domain using deep complex-valued cnn. In : Proceedings of the 30th ACM International Conference on Multimedia. pp. 1474–1483 (2022)

[151] Zheng, X., Qiao, X., Cao, Y., Lau, R.W. : Content-aware generative modeling of graphic design layouts. ACM Transactions on Graphics (TOG) **38**(4), 1–15 (2019)

[152] Zhu, H., Koniusz, P. : Simple spectral graph convolution. In : International conference on learning representations (2020)

[153] Zhu, Y., Lu, S., Zheng, L., Guo, J., Zhang, W., Wang, J., Yu, Y. : Texygen : A benchmarking platform for text generation models. In : The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval. pp. 1097–1100 (2018)

[154] Zhu, Y., Vulić, I., Korhonen, A. : A systematic study of leveraging subword information for learning word representations. In : Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, Volume 1 (Long and Short Papers). pp. 912—-932 (2019)

[155] Zhu, Y., Kiros, R., Zemel, R., Salakhutdinov, R., Urtasun, R., Torralba, A., Fidler, S. : Aligning books and movies : Towards story-like visual explanations by watching movies and reading books. In : Proceedings of the IEEE international conference on computer vision. pp. 19–27 (2015)