

Classification de texte : Approches et avancées en apprentissage profond

Hadjer MEZIANI*, Sabah KIRAT^a

^aCentre de Recherche sur l'Information Scientifique et Technique (CERIST), Alger 16030, Algérie

Résumé

Ces dernières années, le nombre de documents et de textes complexes a connu une croissance exponentielle, nécessitant une compréhension approfondie des méthodes d'apprentissage automatique pour classer les textes avec précision dans de nombreuses applications. De nombreuses approches d'apprentissage automatique ont obtenu des résultats remarquables dans le domaine du traitement du langage naturel.

La classification de texte consiste à classer automatiquement un ensemble de documents dans plusieurs catégories prédéfinies en fonction de leur contenu et de leur sujet. L'objectif principal de la classification de texte est de permettre aux utilisateurs d'extraire des informations des ressources textuelles et de traiter conjointement des processus tels que la recherche, la classification et les techniques d'apprentissage automatique afin de classer différentes catégories.

Le succès de ces algorithmes d'apprentissage repose sur leur capacité à comprendre des modèles complexes et des relations non linéaires dans les données. Cependant, identifier des structures, des architectures et des techniques adaptées pour la classification de texte constitue un défi pour les chercheurs.

Dans cet article, un aperçu des algorithmes de classification de texte est présenté. Cet aperçu couvre différentes méthodes d'extraction des caractéristiques textuelles, des techniques de réduction de la dimensionnalité, des algorithmes et techniques existants. Enfin, les différentes méthodes de classification de texte par apprentissage profond sont comparées et résumées.

Keywords : Classification de texte ; Deep Learning ; TALN

* Corresponding author. Tel.: +213-(0)23-255-416; fax: +213(0)23-255-410
E-mail address: hmeziani@cerist.dz.

1. Introduction

La classification de texte est un problème classique en traitement automatique du langage naturel. La tâche consiste à attribuer des catégories prédéfinies à une séquence de textes donnée. Ces dernières années, l'étude de la classification de texte est devenue de plus en plus importante en raison de la croissance rapide des réseaux sociaux, des blogs et des forums, ainsi que de l'augmentation de la taille des bibliothèques académiques en ligne. En conséquence, la classification de texte est largement utilisée dans les systèmes de recherche d'information et les applications de moteurs de recherche.

Par ailleurs, la classification de texte peut également être utilisée pour le filtrage des spams dans les e-mails et les sms. La plupart des techniques de classification de texte incluent l'extraction de caractéristiques textuelles, la réduction de données, le choix d'un modèle de deep learning et l'évaluation du modèle. De plus, les systèmes de classification de texte peuvent classer le texte selon sa taille, comme au niveau du document, du paragraphe, de la phrase et de la clause (Kamran et al, 2019).

Avant que le deep learning ne devienne le modèle dominant, l'apprentissage automatique traditionnel avait une large gamme d'applications en classification de texte, telles que l'utilisation de techniques d'apprentissage par ensemble comme le boosting et le bagging pour la classification et l'analyse de texte (Naoki et Hiroshi, 1998). En parallèle, (wei et al, 2017) a utilisé une simple régression logistique pour classer les informations textuelles dans le cadre de la recherche d'informations. (Ray, 2010) utilise un classificateur bayésien naïf pour classer des documents, car le Naïve Bayes utilise moins de mémoire et de calcul, et est le classificateur le plus souvent utilisé en apprentissage automatique traditionnel.

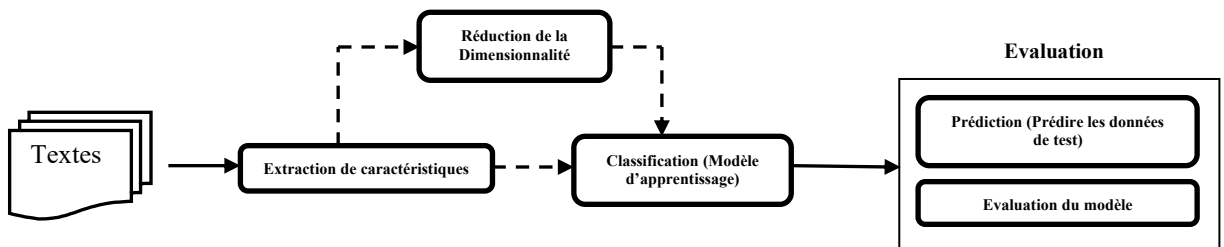


Fig.1. Pipeline de la classification de texte (kamran et al, 2019)

2. Word embedding

(Tomas et al, 2013) Le Word Embedding est une technique clé dans le processus d'extraction de caractéristiques pour la classification de texte. Bien que l'utilisation d'un tokenizer avant la tâche d'extraction pour diviser les phrases en mots et compter le nombre d'occurrences de chaque mot, ainsi que pour générer des représentations syntaxiques des mots, ce processus ne capture pas les informations sémantiques entre les mots. Ce problème peut entraîner de sérieuses difficultés dans la compréhension des informations sémantiques par le modèle. Par exemple, le modèle n-gram ne parvient pas à identifier les similarités entre les mots. C'est pourquoi des chercheurs de Google, ont résolu ce problème grâce au Word Embedding sous forme de vecteurs.

(Derry et al, 2019) Les modèles utilisés pour les word embeddings sont variés, et parmi eux figure le modèle Word2Vec.

Word2Vec représente les mots sous forme de vecteurs, en se basant sur plusieurs caractéristiques telles que la taille de la fenêtre contextuelle et les dimensions des vecteurs.

Word2Vec permet une implémentation efficace des architectures Continuous Bag of Words (CBOW) et Skip-Gram pour calculer les représentations vectorielles des mots. Ces représentations peuvent être utilisées

dans diverses tâches de traitement du langage. Les architectures CBOW et Skip-Gram sont illustrées à la Figure 2.

2.1. Continuous Bag of Words (CBOW)

(Tomas et al, 2013) Une approche consiste à considérer $\{ "Le", "médecin", "le", "patient" \}$ comme un contexte et, à partir de ces mots, prédire ou générer le mot central *"ausculte"*. Ce type de modèle est appelé *Continuous Bag of Words (CBOW)*.

Le modèle CBOW est une méthode populaire pour entraîner des word embeddings, qui sont des représentations des mots dans un espace vectoriel numérique. L'objectif du modèle CBOW est de prédire un mot cible à partir des mots du contexte dans une phrase (Tomas et al, 2013).

« L'architecture CBOW prédit les mots actuels en fonction du contexte. »

2.2. SKIP GRAM

Une autre approche consiste à créer un modèle tel que, donné le mot central *"ausculte"*, le modèle puisse prédire ou générer les mots environnants *"Le", "médecin", "le", "patient"*. Ici, nous appelons le mot *"ausculte"* le contexte. Ce type de modèle est appelé Skip-Gram. Cela contraste avec le modèle CBOW, qui, lui, prédit le mot central à partir des mots du contexte dans une phrase (Tomas et al, 2013).

Le modèle Skip-Gram est essentiellement un réseau de neurones peu profond composé d'une couche d'entrée, d'une couche d'embedding et d'une couche de sortie. L'objectif du modèle est de produire un vecteur de distribution de probabilités en sortie à partir d'un mot cible donné en entrée. Ce vecteur de probabilités (dont la somme est égale à 1) représente la probabilité pour chaque mot d'apparaître dans la fenêtre de contexte du mot cible. L'intérêt est de maximiser les probabilités pour les mots partageant le même contexte et de les minimiser pour les mots qui n'y appartiennent pas (Tomas et al, 2013).

« L'architecture Skip-Gram prédit les mots entourant le mot donné. »

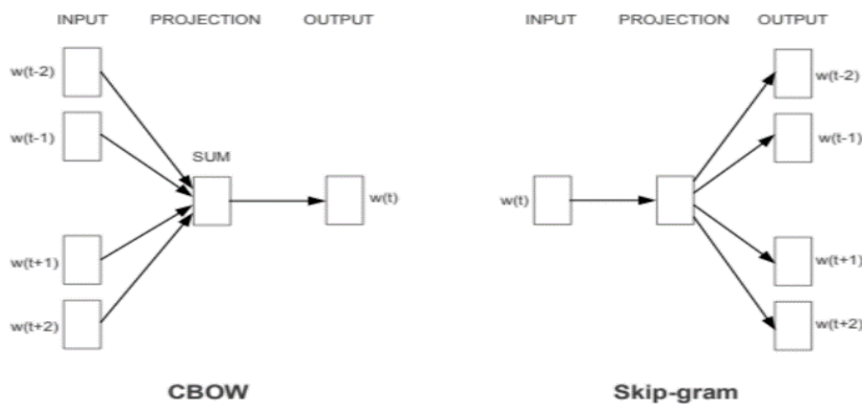


Fig.2.Architectures CBOW et Skip-gram (Tomas et al, 2013).

3. Réduction de dimensionnalité

La réduction de dimensionnalité est une technique fondamentale en apprentissage automatique qui simplifie les ensembles de données en réduisant le nombre de variables ou de caractéristiques en entrée. Cette simplification est essentielle pour améliorer l'efficacité computationnelle et les performances des modèles, en particulier lorsque les ensembles de données augmentent en taille et en complexité. La réduction de dimensionnalité transforme les données en un espace plus simple et de plus faible dimension, tout en conservant leurs principales caractéristiques. Cela facilite les calculs et réduit le risque de sur-apprentissage.

3.1. Analyse en Composantes Principales (PCA)

L'idée principale de la PCA est de réduire la dimensionnalité en trouvant des sous-espaces approximatifs de la distribution des données. Les caractéristiques de dimension n sont projetées par la PCA dans un espace de dimension k , qui correspond à une nouvelle caractéristique orthogonale, également appelée *composante principale*, reconstruite à partir des caractéristiques originales de dimension n . Ces caractéristiques de dimension k sont étroitement liées aux données elles-mêmes (Sasan et al, 2009).

Le premier nouvel axe est choisi comme étant la direction présentant la plus grande variance dans les données originales, le second axe est la direction ayant la plus grande variance dans le plan orthogonal au premier axe, et le troisième axe est celui ayant la plus grande variance dans le plan orthogonal aux 1er et 2e axes. Par analogie, on peut obtenir n axes de cette manière. Enfin, en décomposant la matrice de covariance des données par décomposition en valeurs propres ou par décomposition SVD, on obtient le vecteur propre correspondant aux K plus grandes valeurs propres de la dimension souhaitée, et on le multiplie avec la matrice de données originale pour obtenir les caractéristiques réduites en dimension (Sasan et al, 2009).

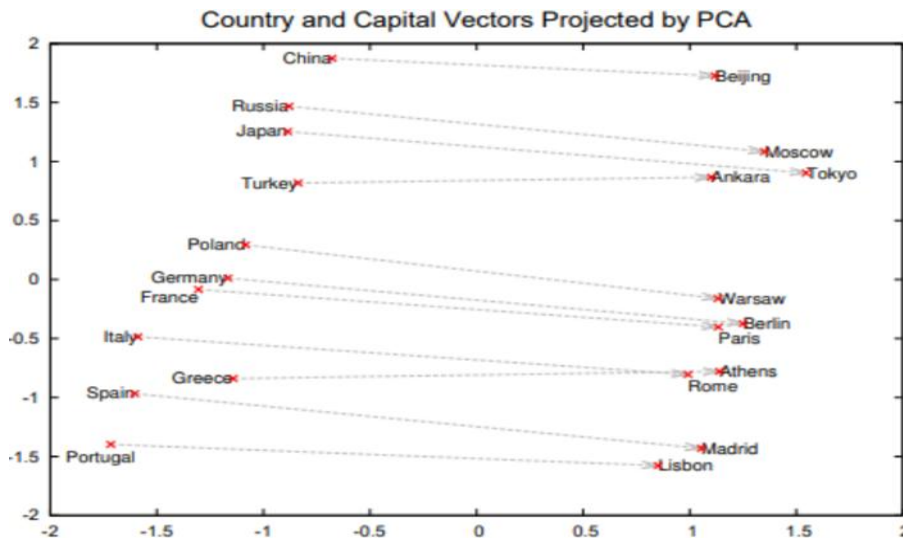


Fig.3. Projection bidimensionnelle par PCA des vecteurs *Skip-gram* de 1000 dimensions représentant des pays et leurs capitales (Sasan et al, 2009).

3.2. Analyse Discriminante Linéaire (LDA)

LDA se distingue de la technique non supervisée de PCA par le fait qu'il s'agit d'une méthode de réduction de dimensionnalité supervisée, ce qui signifie que chaque échantillon du jeu de données est associé à une catégorie (Herauld et al, 1984).

L'idée principale de LDA est de projeter les données dans une dimension inférieure tout en s'assurant que les points projetés de chaque catégorie restent proches les uns des autres et que la distance entre les centres des différentes catégories soit aussi grande que possible (Herauld et al, 1984).

Comme plusieurs catégories sont projetées dans un espace de faible dimension, cet espace projeté n'est pas une simple ligne, mais un hyperplan. Supposons que nous projetions un espace de dimension réduite $\mathbf{d}$, correspondant à un vecteur de base $(w_1, w_2, \dots, w_n)$ (Herauld et al, 1984). La formule principale de LDA est la suivante : S_w est définie comme la matrice de dispersion intra-classe et S_b comme la matrice de dispersion inter-classe, donc la fonction d'optimisation de l'algorithme de réduction de dimension de LDA est formulée comme suit.

$$\underset{W}{\operatorname{argmax}} J(W) = \frac{\prod_{diag} W^T S_b W}{\prod_{diag} W^T S_w W} \quad (1)$$

$$S_b = \sum_{j=1}^k N_j (\mu_j - \mu)(\mu_j - \mu)^T \quad (2)$$

$$S_w = \sum_{j=1}^k S_{wj} = \sum_{j=1}^k \sum_{x \in X_j} (x - \mu_j)(x - \mu_j)^T \quad (3)$$

4. Modèles du deep learning

Les modèles du deep learning ont atteint des résultats à la pointe dans de nombreux domaines et surpassent, à bien des égards, les algorithmes d'apprentissage automatique traditionnels. Ces derniers nécessitent un processus d'extraction de caractéristiques et de sélection de classificateurs, ce qui augmente le coût en main-d'œuvre.

En revanche, les modèles de deep learning sont des modèles de bout en bout pour de nombreuses tâches en vision par ordinateur et en traitement du langage naturel. Dans de nombreuses tâches, les modèles de deep learning présentent des capacités de généralisation et d'ajustement bien meilleures que les algorithmes d'apprentissage automatique traditionnels.

Nous introduirons ici les modèles de base les plus courants en deep learning, tels que *LSTM* (Sepp et Jürgen, 1997), *GRU* (Tomas et al, 2013) et *Transformer* (Ashish et al, 2018), principalement utilisés dans les tâches de classification de texte, ainsi que des modèles de pointe tels que *ELMO*, *GPT*, *BERT*, *XL-Net*, et d'autres modèles de deep learning à la pointe.

4.1. LSTM et GRU

Les réseaux de neurones récurrents (RNNs) sont les architectures de réseaux neuronaux les plus couramment utilisées pour l'extraction et la classification de données textuelles, en particulier pour les données séquentielles telles que les informations textuelles. Ainsi, les RNNs présentent des avantages pour la classification de texte, de chaînes de caractères et de données séquentielles, en tenant compte des informations des nœuds précédents dans un ordre temporel (Sepp et Jürgen, 1997) (Tomas et al, 2013).

4.1.1. Long Short Time memory (LSTM)

LSTM est un modèle amélioré de RNN, qui, comparé au RNN d'origine, surmonte mieux le problème de disparition du gradient grâce à des structures telles que les portes d'entrée, les cellules de mémoire, les portes d'oubli, les portes de sortie et les unités cachées, maintenir ainsi la dépendance à long terme (Sepp et Jürgen, 1997).

Comme le montre la Figure 4, la porte d'oubli contrôle si l'information de la mémoire de la cellule de l'étape précédente est transmise à l'étape actuelle, tandis que la porte d'entrée contrôle comment l'information d'entrée de l'étape actuelle s'écoule dans la cellule de mémoire de l'étape actuelle à travers la cellule de mémoire candidate (Sepp et Jürgen, 1997).

Cette conception permet de faire face au problème de décadence du gradient dans les réseaux neuronaux cycliques et de mieux capturer la dépendance sur de grandes distances temporelles dans les séries chronologiques (Sepp et Jürgen, 1997).

Limites

Les LSTMs peuvent avoir un impact sur les performances et l'évolutivité telles que : Calcul lourd, Mémoire élevée et temps d'entraînement plus long, Difficulté avec les séquences très longues et Manque de parallélisation.

4.1.2. Gated Recurrent Unit

Gated Recurrent Unit (GRU) est un modèle de réseau neuronal, construit en 2015, basé sur une structure empilée, utilisant une structure arborescente pour capturer les informations à dépendance longue durée. *GRU* possède de nombreuses variantes (Jiang, 2024).

GRU est une variation très efficace du réseau *LSTM*. Comparée au *LSTM*, elle ne comporte que deux unités de portes : **la porte de mise à jour** et **la porte de réinitialisation**.

- **La porte de mise à jour** détermine comment les séquences d'entrée et l'état caché précédent influencent l'état caché actuel.
- **La porte de réinitialisation** détermine quelle quantité d'informations de l'état caché précédent peut être oubliée, ce qui permet au modèle d'ignorer certains messages antérieurs non importants.

En résumé, (Chung et al, 2014) (Cho et al, 2014) la structure du *GRU* est plus simple que celle du *LSTM*, et l'entraînement d'un *GRU* est généralement moins coûteux en temps. C'est pourquoi le *GRU* est également un modèle très populaire pour les tâches de classification de texte.

Comme le montre la Figure 4, l'état de l'étape temporelle précédente est supprimé lorsque l'élément de la porte de réinitialisation est proche de 0, et l'état caché de l'étape temporelle précédente est conservé lorsque l'élément de la porte de réinitialisation est proche de 1.

La porte de mise à jour contrôle comment l'état caché doit être mis à jour par l'état caché candidat qui contient des informations sur l'étape temporelle actuelle.

Enfin, l'état caché de l'étape temporelle actuelle est une combinaison de la porte de mise à jour de l'étape actuelle et de l'état caché de l'étape temporelle précédente. La structure du GRU est la suivante (Tomas et al, 2013)

Limites:

Mécanisme de porte moins complexe : La structure simplifiée peut ne pas capturer les schémas subtils et complexes

Risque de surapprentissage : Moins de paramètres rendent les GRU plus compacts, mais ils peuvent être surajustés lorsqu'ils sont entraînés sur de petits ensembles de données sans techniques de régularisation.

Interprétabilité limitée : Les GRU agissent comme des boîtes noires. Comprendre pourquoi certaines prédictions sont faites peut s'avérer complexe, surtout dans les domaines critiques.

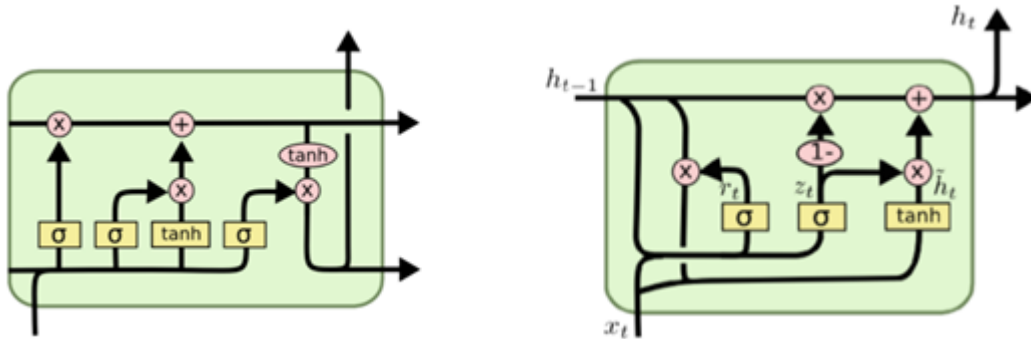


Fig.4. La cellule de LSTM et GRU (Tomas et al, 2013).

4.2. ELMo

ELMo (Matthew et al, 2018) est un modèle de pré-entraînement basé sur un *LSTM* bidirectionnel. Dans les travaux précédents sur word2vec, chaque mot correspondait à un seul vecteur de mot, ce qui n'est pas utile pour les mots polysemes ayant une information sémantique complexe. Par conséquent, dans *ELMo*, le modèle pré-entraîné n'est plus simplement une correspondance de vecteurs.

Lors de l'utilisation d'*ELMo*, une phrase ou un paragraphe est saisi dans le modèle, et le modèle déduit le vecteur de mot pour chaque mot en fonction du contexte.

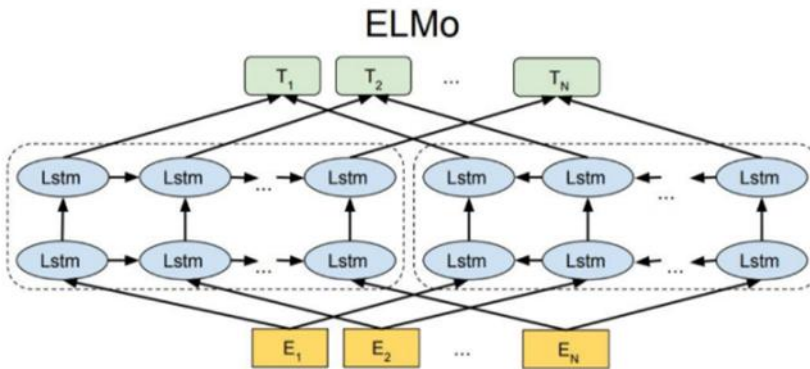


Fig.5. Le modèle ELMo (Matthew et al, 2018)

Le modèle de langage *LSTM* bidirectionnel utilisé par *ELMo* combine les formules en avant et en arrière pour maximiser les probabilités de vraisemblance maximale en avant et en arrière. Ainsi, le modèle d'*ELMo*, présente deux améliorations par rapport au *LSTM* original : la première est l'utilisation d'un *LSTM* multicouche et la seconde est l'ajout d'un modèle de langage en arrière. Pour le modèle de langage en arrière, il est similaire au modèle de langage en avant en ce sens que le contexte arrière est donné pour prédire le contexte avant.

Lorsque le modèle d'*ELMo* termine la tâche de pré-formation, il peut être utilisé pour d'autres tâches de traitement du langage naturel (TALN), ou le *LSTM* bidirectionnel peut être affiné et la représentation de la tâche peut être améliorée. C'est une méthode d'apprentissage par transfert qui peut être utilisée pour des tâches de classification de texte en TALN.

Limites:

L'une des limites d'*ELMo* réside dans son intensité de calcul, car son apprentissage et son utilisation efficace nécessitent des ressources importantes. De plus, il ne capture pas toujours le contexte le plus pertinent pour certains domaines ou tâches spécialisés.

4.3. Transformer

"L'attention est tout ce dont vous avez besoin." L'architecture du modèle Transformer, qui a récemment suscité un grand intérêt parmi les chercheurs, repose sur un modèle d'attention intégrale et est efficace dans les domaines du langage, de la vision et de l'apprentissage par renforcement. En particulier dans le domaine du langage naturel, le modèle Transformer a dépassé les RNN et a obtenu des résultats de pointe dans plusieurs tâches NLP. L'élément le plus important du modèle Transformer est le mécanisme d'auto-attention, qui peut

être vu comme un biais d'induction semblable à un graphe, reliant tous les marqueurs d'une séquence via des opérations de regroupement basées sur l'association (Ashish et al, 2018).

Sur la base de l'auto-attention, les chercheurs ont proposé un mécanisme d'attention multi-têtes, un réseau de feed-forward positionnel, des modules de normalisation par couche et des connecteurs résiduels. L'entrée dans le modèle Transformer est souvent un tenseur de forme (taille de lot, longueur de la séquence). L'entrée passe d'abord par une couche d'encodage qui convertit chaque représentation one-hot de token en un encodage de dimension d , puis un encodage positionnel est ajouté au nouveau tenseur mentionné (Ashish et al, 2018).

Limites

Les transformers peuvent être coûteux en termes de calcul et surajuster les données d'entraînement.

Les transformers peuvent avoir des difficultés à modéliser les dépendances à long terme dans les séquences.

Les transformers s'appuient sur des mécanismes d'attention pour déterminer les parties les plus pertinentes de la séquence d'entrée. Cependant, ces mécanismes peuvent parfois être biaisés, ce qui conduit à des résultats sous-optimaux.

4.4. Generative Pre-Training (GPT)

L'une de ces avancées dans le domaine du traitement du langage naturel est le développement de GPT (Han et al, 2021). GPT est devenu célèbre après le lancement de ChatGPT par OpenAI, une société de recherche (Intro, 2023) spécialisée dans le développement de technologies d'IA. GPT est un modèle d'apprentissage profond pré-entraîné sur de vastes corpus de données textuelles et peut être ajusté pour des tâches spécifiques telles que la génération de langage, l'analyse de sentiment, la modélisation du langage, la traduction automatique, la génération de texte et la classification de texte.

L'architecture du modèle GPT repose sur le modèle Transformer. Le modèle Transformer utilise des mécanismes d'auto-attention pour traiter des séquences d'entrée de longueur variable, ce qui le rend particulièrement adapté aux tâches de NLP. GPT simplifie cette architecture en remplaçant les blocs encodeur-décodeur par des blocs décodeurs. Le modèle GPT reprend l'architecture Transformer et la pré-entraîne sur de grandes quantités de données textuelles en utilisant des techniques d'apprentissage non supervisé. Le processus de pré-entraînement consiste à prédire le mot suivant dans une séquence à partir des mots précédents, une tâche connue sous le nom de modélisation du langage. La modélisation du langage est le

processus qui permet au modèle de découvrir les relations statistiques entre les mots et leur contexte dans les données d'entraînement (Alec, 2018).

Ce processus permet au modèle d'apprendre des représentations du langage naturel qui peuvent être ajustées pour des tâches spécifiques en aval telles que la classification ou la génération de texte (Hou et Ji, 2023). Le modèle est entraîné sur un ensemble de données plus petit, propre à la tâche spécifique lors de l'ajustement fin (fine-tuning), et ses paramètres sont modifiés afin d'optimiser ses performances pour cette tâche (Zhang et al, 2023).

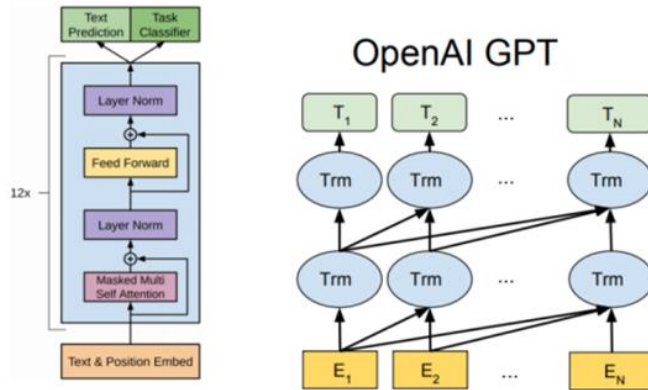


Fig.6.Le modèle GPT

Limites

Les modèles GPT présentent des problèmes tels que les hallucinations, la difficulté à effectuer un raisonnement logique et les contraintes liées à la fenêtre de contexte rendent les modèles GPT (et l'architecture Transformer sur laquelle ils sont basés) peu adaptés à certains cas d'usage.

4.5. Représentation d'encodeur bidirectionnel à partir de Transformer (BERT)

BERT (Devlin et al, 2019) a été introduit par Devlin et al. pour développer un modèle de langage robuste. Avant le succès de BERT, la plupart des approches à la pointe de la technologie (par exemple, CoVe (MacCan et al, 2017), ELMo et GPT) étaient unidirectionnelles ou séquentielles, et ne pouvaient donc pas exploiter les relations contextuelles à travers les entrées textuelles, ce qui entraînait un modèle pré-entraîné de faible qualité. La propriété bidirectionnelle du modèle BERT lui permet d'apprendre le contexte du texte non étiqueté à la fois depuis la gauche et la droite, ce qui donne un bon modèle de traitement automatique du langage naturel pré-entraîné.

BERT est essentiellement la partie encodeur d'un transformeur, avec l'étape supplémentaire de masquer certains des jetons d'entrée et l'objectif additionnel de prédire le mot masqué en se basant uniquement sur le contexte des jetons environnants. L'encodeur reçoit chacun des jetons masqués, conditionnés par les jetons restants dans la phrase, ce qui rend le décodeur superflu (Jacob et al, 2018).

La représentation d'entrée de BERT se compose de trois parties : *WordPiece Token Embedding*, *Position Embedding* et *Segment Embedding*.

Il existe deux jetons spéciaux dans BERT : le jeton [CLS] et le jeton [SEP]. Le jeton [CLS] est toujours le premier jeton de la séquence d'entrée pour les tâches de classification, et le jeton [SEP] sert de séparateur entre les séquences. La longueur maximale d'une entrée, incluant ces deux jetons spéciaux, est limitée à 512.

BERT surmonte les limitations et les inconvénients d'ELMo et d'OpenAI GPT en utilisant les objectifs de pré-entraînement MLM et NSP. Le MLM (Modèle de Langage Masqué) masque aléatoirement certains mots dans les textes d'entrée pendant l'entraînement, puis les prédit à partir des informations contextuelles (Jacob et al, 2018).

De plus, BERT utilise NSP (Next Sentence Prediction) pour prédire si une paire de phrases est séquentielle (c'est-à-dire si la deuxième phrase est bien la suivante dans le texte original)

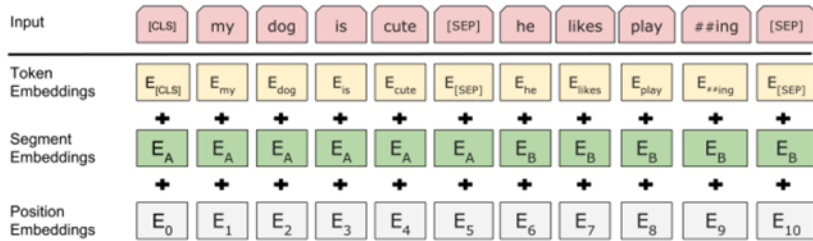


Fig.7.L'encodage de BERT

BERT (Jacob et al, 2018) est très similaire à GPT dans le sens où il s'agit également d'un modèle d'entraînement en deux étapes basé sur le Transformer, divisé en phases de pré-formation et de Fine-Tuning. Les paramètres de ce modèle sont ajustés pour l'adapter à différentes tâches en aval.

Cependant, GPT utilise un Transformer unidirectionnel, tandis que BERT utilise un Transformer bidirectionnel, ce qui signifie qu'aucune opération de masquage n'est nécessaire. De plus, BERT utilise l'encodeur du modèle Transformer, alors que GPT utilise le décodeur, ce qui entraîne des méthodes de pré-formation différentes pour les deux modèles.

Pour distinguer deux phrases, BERT ajoute un encodage de segment qui est appris pendant la pré-formation, en plus de l'encodage positionnel. Ainsi, l'entrée de BERT se compose d'un vecteur de mots, d'un vecteur de position et d'un vecteur de segment qui sont ajoutés ensemble. De plus, les deux phrases sont distinguées l'une de l'autre à l'aide des balises <SEP>.

La phase Fine-Tuning de BERT n'est pas très différente de celle de GPT. En raison du Transformer bidirectionnel, l'objectif d'entraînement auxiliaire utilisé par GPT lors de la phase de Fine-Tuning, c'est-à-dire le modèle de langage, a été abandonné. De plus, le vecteur de sortie pour la prédiction de classification a été modifié : il passe de la position du dernier mot dans GPT à la position de <CLS> au début de la phrase dans BERT.

Limites:

Les modèles BERT s'appuient principalement sur des données textuelles à usage général lors de la phase de pré-apprentissage, tandis que dans un domaine spécifique, le jargon et les contextes spécifiques peuvent être nombreux, ce qui nécessite un apprentissage et une adaptation supplémentaires du modèle.

De plus, les modèles BERT, peuvent rencontrer des goulots d'étranglement lors du traitement de tâches textuelles longues.

4.6.XL-Net

XL-Net (Zhilin et al, 2019), combine les idées de deux modèles pré-entraînés, BERT et GPT, pour proposer un modèle d'apprentissage profond de pointe dans le domaine du traitement du langage naturel. Comme vu précédemment, GPT est un modèle de langage auto-régressif typique, qui présente l'inconvénient de ne pas pouvoir utiliser simultanément les informations contextuelles situées avant et après un mot. BERT,

en revanche, appartient au modèle de langage d'auto-encodeur : il masque aléatoirement une partie des mots dans la séquence d'entrée, puis une des principales tâches du processus de pré-formation est de prédire les mots masqués en se basant sur le contexte.

Par conséquent, XL-Net doit améliorer BERT car la première phase de pré-formation utilise le mode d'entraînement introduisant des marqueurs [Mask] pour masquer certains mots, tandis que la phase d'affinage ne voit pas ces marqueurs de masquage forcé, ce qui crée une incohérence entre les deux étapes pouvant entraîner une perte de performance. De plus, BERT suppose durant la pré-formation que les mots masqués sont conditionnellement indépendants, alors qu'en réalité, il peut y avoir des relations entre ces mots, un aspect pris en compte par XL-Net (Zhilin et al, 2019).

Basé sur le modèle auto-régressif, XL-Net introduit l'objectif d'entraînement du modèle de langage par permutation dans la phase de pré-formation. Par exemple, si la séquence est x_1, x_2, x_3, x_4 et que le mot à prédire est x_3 , le contexte précédent est x_1, x_2 . Afin de prendre en compte le contenu du contexte suivant, une opération de permutation aléatoire génère des séquences comme x_4, x_2, x_3, x_1 qui sont ensuite entrées dans le modèle. Cette approche permet à XL-Net de prendre en compte à la fois le contexte et le contenu grâce à l'opération de masquage dans l'attention du Transformer (Zhilin et al, 2019).

Limites:

Sa complexité architecturale, notamment le PLM (Permutation Language Modeling) et le mécanisme d'attention à deux flux, le rend plus complexe à implémenter et à entraîner. Ce surcroît de complexité se traduit par un coût computationnel plus élevé, tant pour le pré-entraînement que pour l'inférence.

5.Conclusion

La classification de texte est un domaine clé du traitement du langage naturel (TALN), où l'objectif est de regrouper des documents ou des phrases en catégories prédéfinies, comme l'analyse des sentiments, la détection de spams, ou la classification de sujets. Les méthodes basées sur l'apprentissage profond ont révolutionné ce domaine, surpassant largement les approches traditionnelles en raison de leur capacité à apprendre des représentations complexes et des relations contextuelles dans les textes.

Dans les tâches de traitement du langage naturel, la classification de texte basée sur l'apprentissage profond est une direction de recherche très importante. Les progrès rapides de l'apprentissage profond ont largement remplacé les méthodes traditionnelles d'apprentissage automatique.

Dans cet article, nous présentons brièvement certaines méthodes de réduction de caractéristiques appliquées à la classification de texte. Nous nous concentrons sur les modèles de classification de texte basés sur l'apprentissage profond. Ces modèles basés sur l'apprentissage profond sont devenus incontournables dans les tâches de traitement du langage naturel. Ils ont été affinés et ont donné d'excellents résultats dans le domaine de la classification de texte.

Références

- Kamran Kowsari, Kiana Jafari Meimandi, Mojtaba Heidarysafa, Sanjana Mendu, Laura E. Barnes, and Donald E. Brown, 2019. « *Text classification algorithms: A survey* ». CoRR, abs/1904.08067, 2019. URL <http://arxiv.org/abs/1904.08067>.
- Naoki Abe and Hiroshi Mamitsuka, 1998. "Query learning strategies using boosting and bagging". In Jude W. Shavlik, editor, Proceedings of the Fifteenth International Conference on Machine Learning (ICML 1998), Madison, Wisconsin, USA, July 24-27, 1998, pages 1–9. Morgan Kaufmann, 1998.
- Wei Chen, Xiaoshen Xie, Jiale Wang, Biswajeet Pradhan, Haoyuan Hong, Dieu Tien Bui, Zhao Duan, and Jianquan Ma, 2017. « *A comparative study of logistic model tree, random forest, and classification and regression tree models for spatial prediction of landslide susceptibility.* » CATENA, 151:147–160, 2017. ISSN 0341-8162. doi:<https://doi.org/10.1016/j.catena.2016.11.032>. URL <http://www.sciencedirect.com/science/article/pii/S0341816216305136>.

- Ray R. Larson, 2010. "Introduction to information retrieval". J. Assoc. Inf. Sci. Technol., 61(4):852–853, 2010. doi:10.1002/asi.21234. URL <https://doi.org/10.1002/asi.21234>.
- Derry Jatnika, Moch Arif Bijaksana, Arie Ardiyanti Suryania, 2019. «Word2Vec Model Analysis for Semantic Similarities in English Words» 4th International Conference on Computer Science and Computational Intelligence 2019 (ICCSCI), 12-13 September 2019
- Sasan Karamizadeh, S. Abdullah, A. A. Manaf, Mazdak Zamani, and Alireza Hooman, 2009. «Principal component analysis». In Encyclopedia of Biometrics, 2009.
- Hérault, J and Ans. B, 1984. « Réseau de neurones à synapses modifiables: décodage de messages sensoriels composites par apprentissage non supervisé et permanent. ».
- Sepp Hochreiter and Jürgen Schmidhuber, 1997. « Long short-term memory. Neural Comput. », 9(8):1735–1780, 1997. doi:10.1162/neco.1997.9.8.1735. URL <https://doi.org/10.1162/neco.1997.9.8.1735>.
- Jiang, X, 2024 « Development of Text Classification Methods Based on Deep Learning. » DOI: 10.5220/0012829500004547 Paper published under CC license (CC BY-NC-ND 4.0) In Proceedings of the 1st International Conference on Data Science and Engineering (ICDSE 2024), pages 518-523. ISBN: 978-989-758-690-3
- Chung, J.; Gulcehre, C.; Cho, K.; Bengio, Y. « Empirical evaluation of gated recurrent neural networks on sequence modeling». arXiv 2014, arXiv:1412.3555.
- Cho, K.; Van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. «Learning phrase representations using RNN encoder-decoder for statistical machine translation». arXiv 2014, arXiv:1406.1078.
- Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean, 2013. «Efficient estimation of word representations in vector space». In Yoshua Bengio and Yann LeCun, editors, 1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings, 2013. URL <http://arxiv.org/abs/1301.3781>.
- Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer, 2018. « Deep contextualized word representations». In Proc. of NAACL, 2018.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin, 2017. «Attention is all you need». In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems* 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA, pages 5998–6008, 2017. URL <http://papers.nips.cc/paper/7181-attention-is-all-you-need>.
- Han, X, Zhang, Z, Ding, N, Gu, Y, Liu, X, Huo, Y, Qiu, J, Yao, Y, Zhang, A, Zhang, L et al, 2021. «Pre-trained models: Past, present and future, » AI Open, vol. 2, pp. 225–250, 2021.
- Intro, 2023. «Introducing OpenAI». Available: <https://openai.com/blog/introducing-openai>
- Alec Radford. 2018. «Improving language understanding by generative pre-training».
- Hou. W and Ji. Z, 2023. «Geneturing tests gpt models in genomics» bioRxiv, pp. 2023–03, 2023.
- Zhang, C, Zheng, S, Qiao, Y, Li, C, Zhang, M, Dam, S, K, Thwal, C, M, Tun, Y, L, Huy, L, L et al, 2023. «A complete survey on generative ai (aigc): Is chatgpt from gpt-4 to gpt-5 all you need? » arXiv preprint arXiv:2303.11717, 2023.
- Devlin, J, Chang, M, W, Lee, K, and Toutanova, K, 2019. «BERT: Pre-training of deep bidirectional transformers for language understanding». Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2019, 4171–4186.
- McCann, B, Bradbury, J, Xiong, C, and Socher, R, 2017. «Learned in translation: contextualized word vectors». Proceedings of the 31st International Conference on Neural Information Processing Systems. 2017: 6297-6308.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, 2018. «BERT: pre-training of deep bidirectional transformers for language understanding.» CoRR, abs/1810.04805, 2018. URL <http://arxiv.org/abs/1810.04805>.
- Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. Xlnet, 2019. «Generalized autoregressive pretraining for language understanding». CoRR, abs/1906.08237,