Note méthodologique : preuve de concept

Dataset retenu

Le jeu de données utilisé dans ce projet est intitulé "flipkart_com-ecommerce_sample_1050". Il provient du Projet 6, intitulé "Classifiez automatiquement des biens de consommation", qui repose sur une approche de classification fondée sur le traitement automatique du langage naturel (NLP).

Après nettoyage, le jeu de données comprend 1 050 lignes et 17 colonnes. Les variables principales retenues pour ce projet sont :

- cleaned_description : la description nettoyée du produit (variable d'entrée)
- category : la catégorie cible à prédire
- ainsi que d'autres variables secondaires telles que product_rating, brand, description, retail_price, discounted_price, etc., qui peuvent être mobilisées pour des analyses complémentaires.

L'objectif principal est de classer automatiquement les produits dans leur catégorie à partir de leur description textuelle. Pour cela, le projet s'appuie sur une approche basée sur les modèles de langage, comme par exemple BERT, afin d'extraire des représentations sémantiques pertinentes du texte. Ces vecteurs sont ensuite utilisés dans une méthode de clustering non supervisée.

Le prétraitement du texte a été réalisé à l'aide des bibliothèques NLTK et re, selon les étapes suivantes :

- mise en minuscules des caractères
- suppression de la ponctuation et des caractères non alphanumériques
- suppression des stopwords (mots fréquents sans valeur sémantique)
- lemmatisation à l'aide de WordNetLemmatizer

Ce pipeline a été automatisé à l'aide de la fonction preprocess_text(). Une fois le texte nettoyé, il a été tokenisé avec le tokenizer associé au modèle BERT-base uncased.

Il est à noter qu'aucune ligne n'a été exclue lors de cette phase de préparation. Le jeu de données a ensuite été divisé en deux ensembles distincts : entraînement et test.

Ce dataset se prête particulièrement bien à une démarche de preuve de concept, en raison de sa taille réduite, de son bon étiquetage, et de la richesse sémantique des descriptions. Ces caractéristiques en font un support pertinent pour évaluer la capacité des modèles NLP à générer des représentations vectorielles exploitables dans des tâches de regroupement automatique.

Les concepts de l'algorithme récent

- **Rappel sur l'architecture BERT**

BERT (Bidirectional Encoder Representations from Transformers) est un modèle de langage préentraîné introduit par Devlin et al. en 2018. Il s'appuie sur l'architecture des Transformers, présentée par Vaswani et al. en 2017, qui a marqué un tournant majeur dans le traitement automatique du langage naturel.

L'architecture Transformer repose sur le mécanisme de self-attention, qui permet au modèle d'évaluer l'importance de chaque mot d'une séquence en fonction des autres. Contrairement aux architectures séquentielles classiques comme les RNN ou les LSTM, les Transformers traitent l'ensemble de la séquence en parallèle. Cette particularité améliore non seulement la vitesse d'entraînement, mais aussi la capacité à capter les dépendances à longue distance entre les mots.

L'architecture complète d'un Transformer est composée de deux blocs : un encodeur et un décodeur. Toutefois, BERT utilise uniquement la partie encodeur, qu'il empile en plusieurs couches pour encoder finement le contexte des séquences textuelles.

L'une des forces majeures de BERT est sa capacité à prendre en compte le contexte des mots de manière bidirectionnelle, c'est-à-dire en intégrant à la fois les mots précédents et les mots suivants. Cette approche le distingue des modèles auto-régressifs, qui traitent le texte dans un seul sens. BERT est préentraîné sur deux tâches : le Masked Language Modeling (MLM), où le modèle doit prédire des mots masqués dans une phrase, et la Next Sentence Prediction (NSP), qui évalue la cohérence entre deux phrases consécutives.

Une fois préentraîné, BERT peut être spécialisé pour une tâche spécifique via un fine-tuning, en ajoutant par exemple une simple couche de classification connectée au token [CLS], qui représente la séquence entière.

L'architecture BERT-base comprend 12 couches encodeuses, 768 dimensions cachées et 12 têtes d'attention, pour un total d'environ 110 millions de paramètres. Bien que très performante, cette architecture demeure coûteuse en termes de ressources, notamment lors du fine-tuning complet sur des jeux de données spécifiques.

Sources :

[1] Devlin et al. (2018), BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, arXiv:1810.04805

[2] Vaswani et al. (2017), Attention is All You Need, arXiv:1706.03762

[3] Jason Brownlee (2021), A Gentle Introduction to the Transformer Model, Machine Learning Mastery, https://machinelearningmastery.com/the-transformer-model/

- **Le principe de LoRA (Low-Rank Adaptation of Large Language Models)**

LoRA est une technique introduite par Hu et al. (2021) pour rendre le fine-tuning des grands modèles préentrainés (comme BERT ou GPT) plus efficace et moins coûteux [3]. L'idée clé est de figer les poids du modèle préentrainé et d'ajouter des matrices de faible rang (low-rank) à certains poids clés, typiquement dans les couches d'attention (query et value).

D'un point de vue mathématiques, LoRA repose sur l'idée de ne pas réentraîner directement les matrices de poids complètes du modèle préentraîné, mais d'y ajouter une mise à jour de faible rang. Cette mise à jour s'appuie sur deux matrices de petite taille apprises pendant le fine-tuning.

Au lieu d'optimiser la matrice de poids W (de taille d × k), LoRA conserve W inchangée (elle est « gelée »), et apprend deux matrices :

- **A** : de taille r × d (projection dans un espace réduit)
- **B** : de taille k × r (projection inverse vers l'espace original)

La mise à jour totale devient alors :

W adapté = W + α × B × A

où :

- **W** : matrice de poids d'origine (non modifiée)
- **A** : matrice de réduction (rang faible)
- **B** : matrice de reconstruction
- **α** : facteur de mise à l'échelle, souvent fixé pour stabiliser l'apprentissage
- **r** : rang faible choisi (par exemple 4, 8 ou 16)

Cette méthode permet de n'entraîner qu'un faible nombre de paramètres supplémentaires, tout en obtenant une performance proche (ou supérieure) au fine-tuning complet du modèle.

LoRA présente plusieurs avantages majeurs :

- Réduction de la mémoire GPU utilisée pendant l'entraînement
- Possibilité de fine-tuner de très grands modèles sur des machines légères

- Compatible avec des librairies comme PEFT (Parameter-Efficient Fine-Tuning)

Cette approche est devenue populaire car elle permet de démultiplier les capacités d'adaptation de modèles très puissants (comme BERT) sans les coûts habituels en termes de stockage, de temps de calcul ou de consommation d'énergie.

Sources :

[3] Hu et al. (2021), "LoRA: Low-Rank Adaptation of Large Language Models", arXiv:2106.09685

Hugging Face Blog, "Parameter-Efficient Fine-Tuning with LoRA, PEFT and QLoRA" (https://huggingface.co/blog/peft)

KDNuggets, "LoRA: Fine-Tuning Large Models Efficiently" (https://www.kdnuggets.com/2023/04/lora-fine-tuning-large-models-efficiently.html)

La modélisation

Dans le cadre de cette veille technique, nous avons exploré différentes approches de modélisation autour des modèles de langage préentraînés, en particulier le modèle BERT enrichi par la technique de Low-Rank Adaptation (LoRA). L'objectif principal de cette expérimentation était d'évaluer dans quelle mesure ces outils permettent de structurer automatiquement des descriptions textuelles de produits, en vue d'une application potentielle à la classification ou à la segmentation non supervisée.

Utilisation de BERT comme extracteur de représentations

Dans un premier temps, nous avons utilisé le modèle BERT-base uncased sans phase de fine-tuning. À l'aide du tokenizer associé, nous avons extrait des embeddings textuels à partir du token spécial [CLS], supposé capturer le sens global de chaque description produit. Ces représentations vectorielles ont été utilisées pour :

- une réduction de dimension via t-SNE, afin de visualiser la distribution sémantique des produits ;
- un clustering à l'aide de l'algorithme KMeans pour identifier des regroupements sémantiques.

La qualité des clusters a été évaluée à l'aide de l'indice ARI (Adjusted Rand Index), qui permet de mesurer la cohérence entre les regroupements non supervisés et les catégories réelles.

Fine-tuning du modèle via LoRA

Dans un second temps, nous avons appliqué la technique LoRA afin d'adapter BERT à notre tâche. Cette méthode permet un fine-tuning efficace en réduisant la quantité de paramètres à ajuster. Après entraînement, de nouveaux embeddings ont été générés, puis analysés selon le même protocole (t-SNE, KMeans, ARI). Les résultats ont montré une structuration plus nette et une amélioration significative du score ARI, confirmant l'apport du fine-tuning léger.

Interprétation du modèle avec LIME

Afin d'illustrer les décisions du modèle, nous avons utilisé LIME, qui permet d'identifier les tokens les plus influents dans une prédiction. Cette première approche d'explicabilité a permis de mettre en évidence certains mots caractéristiques de chaque catégorie.

Enjeux identifiés pour un usage opérationnel

- LoRA est une solution efficace et peu coûteuse pour adapter un modèle de type transformer dans des contextes contraints en ressources (GPU, temps).
- Les embeddings BERT peuvent être exploités pour structurer un corpus dans un cadre non supervisé, sans annotation préalable.
- L'intégration d'un outil d'explicabilité comme LIME facilite la compréhension des prédictions par les utilisateurs métier.

Une synthèse des résultats

L'objectif de cette expérimentation était de comparer la qualité des représentations textuelles générées par un modèle BERT classique et par un modèle BERT fine-tuné avec LoRA, en évaluant leur capacité à regrouper des textes similaires dans le cadre d'une approche non supervisée.

Pour ce faire, les embeddings extraits ont été projetés en deux dimensions à l'aide de la méthode t-SNE, puis regroupés à l'aide de l'algorithme de clustering KMeans. La cohérence entre les clusters ainsi formés et les catégories réelles a été mesurée à l'aide de l'indice Adjusted Rand Index (ARI).

Le modèle BERT sans fine-tuning a obtenu un ARI de 0,29, indiquant une faible correspondance entre les regroupements générés automatiquement et les classes cibles. En revanche, après l'application de LoRA pour effectuer un fine-tuning léger du modèle, l'ARI a atteint 0,50, ce qui constitue une amélioration significative de la structuration sémantique des représentations.

Ces résultats montrent qu'un fine-tuning partiel, tel que proposé par LoRA, permet d'aligner les représentations vectorielles avec les catégories pertinentes, tout en évitant le coût d'un réentraînement complet du modèle. Cela confirme l'intérêt de LoRA comme solution efficace pour adapter des modèles de langage à des tâches spécifiques tout en maîtrisant les ressources techniques mobilisées.

L'analyse de la feature importance globale et locale du nouveau modèle

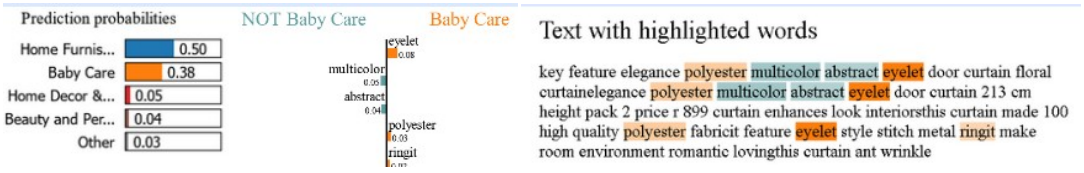
La notion de feature importance est essentielle pour comprendre le comportement d'un modèle de machine learning, en identifiant les caractéristiques qui influencent le plus ses décisions. On distingue généralement deux types d'importance : globale et locale.

L'importance globale mesure l'effet moyen d'une variable sur les prédictions à l'échelle de l'ensemble du jeu de données. Elle permet, par exemple, de déterminer si certaines variables comme l'âge, le revenu, ou certains mots dans un texte sont prédictives dans la majorité des cas. À l'inverse, l'importance locale s'intéresse à une observation spécifique, et permet d'expliquer pourquoi le modèle a pris une décision précise dans ce cas particulier. Elle est particulièrement utile dans les contextes où une justification individuelle est requise, comme l'explication d'un refus de crédit ou la classification d'un produit.

Plusieurs méthodes permettent de mesurer cette importance, notamment les approches intégrées à certains modèles (par exemple les poids dans les arbres de décision ou les coefficients dans une régression), ou des méthodes dites agnostiques comme LIME, qui consistent à perturber les entrées pour observer l'effet sur la sortie du modèle.

Dans le cadre de cette veille, nous avons appliqué la méthode LIME à un modèle BERT fine-tuné avec LoRA, utilisé pour classer des produits en différentes catégories. L'interprétation locale nous a permis de visualiser les mots clés influents dans la décision du modèle pour un produit donné. Par exemple, des termes comme abstract, multicolor orientent la classification vers la catégorie Home Furnishings.

En agrégeant les explications issues de plusieurs exemples, il devient possible d'estimer une forme d'importance globale des mots les plus déterminants sur l'ensemble des prédictions du modèle. Cette approche contribue à mieux comprendre le comportement global du modèle, tout en permettant de vérifier que ses décisions sont cohérentes avec les attentes métier.



Les limites et les améliorations possibles

L'utilisation de BERT combiné à LoRA a permis de réduire le coût du fine-tuning tout en conservant de bonnes performances sur la tâche de classification de textes. Toutefois, ce type de modèle reste relativement lourd, avec une consommation mémoire importante, ce qui peut entraîner des contraintes matérielles, notamment sur le GPU. Cela a notamment été observé lors de l'utilisation de méthodes d'explicabilité comme LIME, qui peuvent s'avérer coûteuses en ressources.

Par ailleurs, le modèle BERT de base demeure généraliste : même après fine-tuning, il peut ne pas saisir toutes les spécificités liées à un domaine ou à un métier donné. Sur le plan de l'interprétabilité, LIME fournit des explications locales intéressantes, mais sa fiabilité peut être limitée par la nature aléatoire des échantillons qu'il génère, ce qui affecte la stabilité des résultats.

De plus, la feature importance calculée ici ne porte que sur les tokens textuels. Cela limite la compréhension du modèle à une seule dimension d'analyse, alors que d'autres variables pourraient également influencer les décisions du modèle.

Pour répondre à ces limites, plusieurs pistes d'amélioration peuvent être envisagées : utiliser des modèles plus légers afin d'optimiser l'usage mémoire, ou encore explorer des modèles préentraînés sur des corpus proches du domaine d'application, comme les descriptions de produits e-commerce.

Du côté de l'interprétabilité, il serait pertinent de compléter l'analyse par une agrégation plus systématique des explications locales fournies par LIME, afin de construire des indicateurs globaux plus robustes. L'importance des mots pourrait également être croisée avec d'autres variables disponibles dans les données (prix, marque, type de produit), ce qui permettrait d'enrichir l'analyse et d'aller vers des modèles à la fois plus efficaces, plus compréhensibles, et mieux adaptés à leur contexte d'usage.