Projet UE5 : Classification des textes issues de Pubmed

Nisrine Bennor

Master 2 Informatique biomédicale
2019-2020

Description des données :

Dans ce projet, nous disposons des données permettant de décrire un ensemble d'articles scientifiques autour de l'intelligence artificielle tirés de Pubmed. Ces données sont regroupées dans un fichier json nommé ai_pub_samp.json. Au total, on a 10000 articles et pour chaque article on a le titre 'title', la catégorie 'categories', l'abstract 'abstract', et plusieurs autres informations comme les auteurs, le journal, l'année, les mots-clés, l'année ...

Objectif du travail :

Nous nous proposons d'entrainer des modèles et de comparer leur performances. Ces modèles ont pour but de classifier les articles dans des catégories spécifiques 'categories' en fonction de leurs titres, abstract, ou les 2 rassemblés.

Méthodes : description des modèles

Dans ce problème de classification de texte, on va utiliser principalement des modèles de réseaux de neurones de convolution (CNN) et des modèles de réseaux de neurones récurrents (RNN). Les techniques classiques de classification de texte comme le TF-IDF sur les ngrammes des mots et des caractères ne sont pas abordées.

❖ Modèles de CNN :

1.Modèle basique: c'est le modèle donné dans le TP, il renferme une couche d'embedding, une couche de dropout, une couche de convolution (nombre de filtres= 32 et taille de filtre=2), une couche de maxpooling, une couche dense de 128 neurones avec activation Relu et le classifieur (Dense et activation 'softmax', puisqu'il s'agit d'un problème de classification multiclass)

2. Modèle basique avec changement des hyperparamètres : le même modèle est repris avec augmentation de nombre de filtres à 256, diminution de 'learning rate' à 10^{-4} et augmentation du nombre d'epoches à 50.

3. Modèle de convolution multi-channel : Ce modèle a été inspiré à partir d'un article accessible via le lien suivant :

https://ahmedbesbes.com/overview-and-benchmark-of-traditional-and-deep-learning-models-in-text-classification.html?fbclid=IwAR0k6eiPCN79sMrTgmnIQUVThz92cB2XERethXD1lcH9tP8tz97Dh nj3zaw

Dans ce modèle, on utilise des CNN multiples parallèlement, on applique des filtres de tailles différentes : ici 2, 3 et 5. Ceci permet au document d'être parcouru avec différentes résolutions ou différents ngrammes (2-grammes, 3-grammes et 5-grammes). Ce modèle a été décrit pour la première fois par Yoon Kim en 2014 dans son papier intitulé : "Convolutional Neural Networks for Sentence Classification."

- Modèles de RNN :

1. Modèle LSTM (Long Short-Term Memory) basique avec changement des hyperparamètres : Dans ce modèle, nous avons repris l'exemple donné au TP, mais on a diminué le dropout à 0.3 et on a supprimé la couche dense de 64.
2. Modèle GRU (bidirectional Gated Recurrent Unit) : Ce modèle est aussi pris du même article. GRU est une variante plus rapide que le LSTM. Le fait qu'il est bidirectionnel lui permet de scanner la séquence des mots dans les 2 directions.

- Modèles combinant RNN et CNN : toujours pris du même article, ce modèle renferme : une couche d'embedding, une couche de dropout, une couche GRU, une couche de convolution, un avgpooling, un maxpooling, une concaténation de avgpooling et de maxpooling et enfin le classifieur.

Tous ces modèles sont appliqués sur :

- Le titre
- L'abstract

Ces modèles sont appliqués avec des embedding non-pré-entrainés dans un premier temps puis avec des embedding pré-entrainés de GloVe, la taille de l'embedding est de 50.

Pour le titre et abstract on a utilisé d'autres modèles combinatoires des modèles basiques (2 CNN, 2 LSTM, 1CNN et 1 LSTM), dans cette partie on a travaillé qu'avec des embedding préentrainés Glove.

Avant d'appliquer les réseaux de neurones, une étape de preprocessing et de transformation des textes en vecteurs numériques est indispensable.

Etapes de preprocessing :

La visualisation du nombre et de la fréquence des différentes catégories dans nos données nous a donné au total 97 catégories, avec des catégories qui sont dominantes et d'autres peu fréquentes. L'idée était de spliter nos données en train et test dans lesquels on retrouve les mêmes répartitions des catégories en utilisant l'option stratify= Y. Pour faire cette opération on a dû écarter les catégories qui ne sont présentes qu'une seule fois et qui sont 5 catégories car l'option stratify ne fonctionne qu'avec un minimum de 2 représentants pour une catégorie. Les articles écartés sont ajoutés aux données de test après le split. Le train représente 80% de nos données. A chaque fois, on enregistre les données du train (x et y) dans un dataframe et les données de test dans un autre dataframe, puis dans un fichier csv dans un dossier « data »

Une fois on a nos données divisés en train et test, on commence à transformer nos variables x et y, pour x c'est-à-dire les titres et les abstracts on a à décomposer en mots

et à créer pour chaque mot un vecteur puis on doit faire un padding. Pour la variable y, on a à créer une sorte de dictionnaire, pour une catégorie correspond un index.

(fichier Projet_data_preparation.ipynb)

Résultats :

1. Classification basée sur les titres (fichier Classification_Title.ipynb)

A – Embedding non-préentrainés

	CNN			RNN		CNN_RNN
Modèle	Basique*	Hyper*	Multi-channel*	LSTM*	GRU*	
Précision	0.3205	0.3705	0.403	0.3265	0.380	0.349

B- Embedding préentrainés :

	CNN			RNN		CNN_RNN
Modèle	Basique*	Hyper*	Multi-channel*	LSTM*	GRU*	
Précision	0.34	0.377	0.402	0.3990	0.395	0.3885

2. Classification basée sur les abstracts(Fichier Classification_Abstract.ipynb)

A – Embedding non-préentrainés

	CNN			RNN		CNN_RNN
Modèle	Basique*	Hyper*	Multi-channel*	LSTM*	GRU*	
Précision	0.38	0.437	0.502	0.2715	0.405	0.3985

B- Embedding préentrainés :

	CNN			RNN		CNN_RNN
Modèle	Basique*	Hyper*	Multi-channel*	LSTM*	GRU*	
Précision	0.388	0.4505	0.46	0.2975	0.461	0.43

3. Classification basée sur les titres et les abstracts (Fichier Classification_Title_Abstract.ipynb)

Dans tous les modèles testés dans cette partie, on a employé les embedding pré entrainés de GloVe de dimension 50.

- CNN sur le titre et CNN sur l'abstract : précision=0.46
- LSTM sur le titre et LSTM sur l'abstract : précision= 0.3795
- LSTM sur les titres et CNN sur l'abstract : précision= 0.4450

- CNN sur les titres et LSTM sur l'abstract : précision= 0.4130

Discussion

La classification des articles en catégories en fonction de leur titres donne le score de précision le plus haut avec le modèle de CNN multi-channel (score=0.403) suivi du modèle GRU (score=0.38). L'utilisation conjointe de GRU et convolution (modèle CNN RNN) n'a pas donné une précision meilleure (score=0.349).

L'utilisation des embedding pré-entrainés GloVe a permis une amélioration des résultats. L'amélioration est plus nette dans le modèle LSTM (passant de 0.3265 à 0.3990). L'utilisation de ces embedding permet de mieux initialiser les poids de notre réseaux ce qui permet d'améliorer la précision du modèle mais cela est en fonction de la performance de ce modèle. (Si celui-ci présente déjà une bonne précision l'amélioration est minime).

En se basant sur l'abstract la classification des articles scientifiques en catégories donne des scores plus élevés atteignant 0.502 avec le modèle CNN multichannel. Le LSTM fonctionne moins bien avec les abstracts, ceci peut être dû à une architecture de réseaux plus adaptées aux titres, l'ajout d'autres couches de neurones avant l'utilisation de classifieur pourrait améliorer la précision. L'utilisation des embedding pré entrainés permet une amélioration des scores dans un certain nombre de modèles mais ne permet pas de dépasser la barre de 0.502 obtenue avec le CNN multichannel.

La modification des hyperparamètres permet d'améliorer la précision mais le problème que ça nécessite plusieurs tests et on doit changer un seul paramètre à la fois pour voire son effet et le sens de sons modification lors du nouveau test. Pour la classification basée sur les titres et les abstracts, le meilleur score a été obtenu en combinant 2 convolutions une sur les titres et une sur les abstracts (précision=0.46).Cette précision dépasse celle de la classification basée sur les abstract (précision=0.4505) et celle basée sur les titres (0.377). En combinant 2 LSTM on a obtenu 0.3795 donc ce modèle n'est pas plus performant que LSTM appliqué sur les titres (0.39).

Conclusion

- Il s'agit d'un problème de classification de texte avec un grand nombre de classes (catégories).
- Plusieurs classes ne sont pas bien représentées dans nos données -> nécessité d'avoir plus de données représentatives de ces classes.
- La précision n'est pas assez bonne (0.5 dans les meilleurs des cas).
- L'abstract parait plus riche et plus adapté à la catégorisation.
- Des modèles plus complexes sur la combinaison des titres et abstracts restent à tester ...
- Les embedding pré entrainés améliorent la précision jusqu'à un certain niveau.

Lien github : https://github.com/NisrineBennor/Classification_text_pubmed1