Analyse du contexte des modèles NLP en français et en anglais.

Ismaïl Oudahya,

***Résumé*—Grâce au traitement automatique des langues, nous avons pu utiliser des modèles comprenant les langages humains dans différentes langues. Dans cet article, nous analyserons les différents comportements des modèles BERT et CamemBERT. Dans un premier cas, nous regarderons leurs résultats du pré-entraînement et ensuite dépendant des résultat, nous leur apporterons des modifications dans les couches de neurones. Cela dit, nous constaterons qu'il est important d'avoir de bon hyperparamètres pour les modèles. Dépendant des paramètres, les modèles peuvent avoir de meilleurs résultats.**

I. Introduction

DEPUIS l'évolution de l'internet il y a eu une hausse de consommation. Les personnes ont commencé à découvrir de nouvelles choses et à partager ce qu'il apprécie ou non. Beaucoup d'opinions, de questionnement, d'information se retrouvent dissimulées un peu partout sur Internet. Par exemple, un avis sur la visite d'un musée, les personnes mettent à jour les notes de Google pour donner leur avis, mais imaginons pour des millions de personnes visitant toute chose différentes tous les jours. Comment classer toutes ces informations ? La réponse se trouve dans l'automatisation. Automatiser la compréhension d'un avis ou d'un questionnement rendrait les choses beaucoup plus simples et permettrait de satisfaire tout le monde.

Comment automatiser la compréhension du langage humain ? Que dire à l'ordinateur pour comprendre les informations ? Pour mieux comprendre ce problème, de nombreuse études ont été menées sur l'analyse du langage que ce soit en français ou en anglais. Cette capacité, à analyser le langage est appelé NLP. Le NLP est une branche de l'étude sur l'intelligence artificielle entre un humain et un ordinateur. Le langage humain est fait pour la communication, le partage de connaissances ou même de sentiment. Le NLP s'est avéré très utile pour la compréhension du langage, car il a un ensemble de technique permettant de collectionner toute la structure de la grammaire et de comprendre le sens d'une phrase.

Bien que cela reste un exploit d'apprendre à un ordinateur la compréhension du langage humain, il reste un problème. Comment interagir entre l'interaction et la précision de chaque sens d'une phrase ? Toute langue est différente et ont une manière de s'exprimer toute différentes. Une solution à ce problème, c'est le machine learning.

Cette technique d'apprentissage permet à la machine d'apprendre toute seule et de manière automatiser. Il consiste à s'entraîner à l'aide de ses connaissances et d'en sortir un modèle.

En 2018, la filière Google AI a publié un modèle de deep-learning pré-entraîné, capable de résoudre plusieurs problèmes de langage NLP, appelé BERT. Il a été entraîné sur plusieurs heures et à coûter près de 60 000 dollars. D'autre ont dérivé du modèle, pour le français, on a eu camemBERT. Ces deux modèles sont complétement open source.

II. Etat de l'art

Différents modèles ont été mis en place un peu partout sur Internet avec différents paramètres. Ces paramètres des modèles peuvent avoir un grand impact sur la sortie des résultats. Ces modèles permettront de trouver réponse à nos questions.

A. Introduction BERT

Lorsqu'on donne une phrase à BERT, il utilise une technique propre à lui appeler MAKED LM. Il masque des mots aléatoirement dans la phrase, et essaye ensuite de les prédire. Le modèle essaye donc de trouver des mots qui correspondent au contexte et pas seulement au remplacement des mots. Il prend en compte les mots qu'il a découverts et les suivantes en même temps, pour construire une meilleure approche de la phrase.

BERT est un type transformers, cela sert à déterminer le sens d'un mot avec toute la phrase et prends une décision en une étape basée sur son analyse. BERT se base sur l'architecture des transformers, qui consiste en un encodeur pour lire le texte et un décodeur pour faire la prédiction. BERT se limite seulement à un encodeur, car son objectif est de créer un modèle du langage NLP.

B. Introduction CamemBERT

Malheureusement, le modèle BERT ne peut gérer qu'une seule langue à la fois, sachant que les prédictions du modèle NLP diffèrent pour chaque langue, leur syntaxe est différente. On se retrouve donc avec d'autres modèles de type BERT. Il existe un modèle en français qui permet

Superviseurs : NACHTEGAEL Charlotte

de maîtriser la grammaire et la syntaxe française et dotée d'un nombre de vocabulaire conséquent.

CamemBERT a été entraîné pendant 17h sur 256 cartes graphique de série Nvidia v100 avec 32 gb de vram chacune.

Ici, le modèle est un peu différent de BERT. Il a un seul objectif : prédire les mots masqués d'une séquence, mais BERT lui, en avait deux. [1]

C. Fine tuning

Le fine tuning est une technique permettant d'optimiser les hyperparamètres d'un modèle. Elle consiste à créer plusieurs apprentissages avec le même modèle, mais avec des hyperparamètres différents. Certains modèles pourront être plus performants ou au contraire inutilisable dépendant des choix fait.

Après un pré-entraînement sur une quantité massive de données, le modèle peut être introduit avec du fine tuning sur une tâche spécifique, ce qui permet d'orienter les résultats du modèle. [2]

Sachant qu'il existe différents modèles qui se ressemble fortement, mais avec des hyperparamètres différent, leur prédiction change-t-elle à cause de la langue ? Nous allons donc fine tuner le modèle BERT et CamemBERT en ajoutant quelque couches de réseau neuronal par nous-mêmes et en gelant les couches actuelles et analyser leur résultats. Le fine tuning sera réalisé avec une carte graphique GeForce GTX 1070 Ti.

D. Type de modèle

Il existe différents types de modèle pour BERT et CamemBERT. Pour BERT, il y a le BERT_base et BERT_large. Bert_base utilisé 12 couches de transformer avec une taille de 768 dimensions cachées et un nombre de 12 têtes d'attention , soit environ 110 millions de paramètres entraînables. Pour Bert_Large, il utilise 24 couches de transformer avec 16 têtes d'attention et possède environ 340 millions de paramètres. Dépendant de leur hyperparamètres, ils seront utilisés pour des structures telle que la classification, question réponse...[3]

Lors du pré-entraînement des modèles, il est nécessaire d'avoir beaucoup de données à fournir au modèle. Ici, l'utilisation de BooksCorpus pour un total de 800 millions de mots et le Wikipédia Anglais 2500 millions de mots ont permis au modèle de s'entraîner.

Du côté de CamemBERT, puisqu'il a été créé à partir de BERT, ils ont les mêmes paramètres. Les données d'entraînement ont été utilisées sous-corpus français OSCAR, Common Crawl nommé CCNet et une copie de Wikipédia française. [4]

III. Méthodologie

Nous réaliserons 2 types de test différents. Le but est d'avoir une idée de si l'output à un bon taux de succès par rapport au français et l'anglais dépendant de leur fine tuning.

1) Analyse du taux de succès MLM avec pré-entraînement.

Lors du pré-entraînement des deux modèles, ils seront souvent déjà aptes à réaliser les prédictions d'une phrase ou d'un contexte, mais c'est tout ce qu'ils peuvent faire. Ils sont très limités dans leur action. Notons que les modèles sont principalement destinés à être utilisés sur des tâches qui utilisent une phrase entière pour prendre une décision. Lors de l'affichage des résultats que les modèles trouveront, un score au pourcentage de réussite sera mis en avant ainsi que le résultat correspondant.

2) Fine tuning pour BERT et CamemBERT.

Comme nous voulons comparer l'analyse d'un contexte pour l'anglais et le français. Nous avons mis en place deux dataset différentes. Pour Bert, une dataset sur la classification des critiques sur les films a été mise en avant. Deux types de résultat prévus pour chaque critique. Lorsque la critique du film est négative un label valant 0 sera posté , 1 si la critique est positive. Pour la partie CamemBERT, une dataset comprenant des Tweets en français a été implémenter dans le modèle. Leur label sont identique aux critiques des films. Le but ici sera de rajouter des couches en plus au modèle pour l'entraîner à reconnaître les critiques, donc analyser les contextes de chaque commentaire et de prendre une décision sur le résultat. Lors du fine tuning, nous allons rajouter 3 couches d'entrée une pour chaque liste :

- L'encodage de chaque mot pour le MLM
- La position des mots dans une phrase
- Un segment représentant des paires de phrases pour des tâches plus spécifique.

Quand les poids passeront dans la couche final, une classification des résultats se fera par le biais d'une fonction d'activation.

Pour finir, une action se fera en une fonction qui permet de minimiser le coût du réseau neuronal. C'est donc un algorithme d'optimisation qui permettra de mettre à jour les poids du réseau de neurones.

Lors de l'entraînement, une epoch et un batch size seront transmis afin de bien structurer la manière dont le réseau s'entraîne. Le batch size est un hyperparamètre qui définit le nombre d'échantillons à traiter avant de mettre à jour les paramètres du modèle. Nous pouvons voir cela comme une boucle dans un langage de programmation. Le nombre d'epochs est un hyperparamètre qui définit le nombre de fois que l'algorithme d'apprentissage va travailler sur l'ensemble des échantillons.

Pour des raisons de manque de matériel, nous avons tenté de prendre une petite epoch et un petit batch size pour ne pas avoir une longue attente d'entraînement.

Voici un tableau représentant les modèles entraîner :

Model	Epochs	Batch-size
Bert	5	32
CamemBERT	3	16

L'entraînement de BERT a pris environ 4h et CamemBERT environ 8h. Le modèle de CamemBERT a pris plus de temps d'entraînement, car le modèle était plus grand en terme de couches que Bert.

A. Fondement mathématiques :

Pour modéliser l'entraînement pour chaque modèle, nous avons dû faire appel à des fonctions d'activation et une fonction d'optimisation.

Les fonctions d'activation sont activées lors de la dernière couche avec les poids sortant des couches précédentes.

Sigmoid :

$$f(x) = 1/1 + e^{-x}$$

Softmax :

$$f(z_i) = exp(z_i)/\sum_j exp(z_j)$$

Dans la fonction sigmoid x, représente les valeurs de sortie dans la couche de sortie. De même pour Softmax.

AdamW :

$$w_t + 1 = w_t - (\alpha_t/(v_t + \epsilon)^{1/2}) * [\delta * L/\delta w_t]^2$$

— $w_t + 1$ = poids en temps $t + 1$
— α_t = taux d'apprentissage en temps t
— v_t =la somme des gradient
— ϵ = une constante en 10^{-8}
— δL = dérivé de la fonction de perte
— δw_t = dérivé des poids en temps t

Durant l'entraînement, nous les appliquons ces formules directement depuis la librairie tensorflow.

B. Instructions pour reproduire l'expérience

Dans le cas de CamemBERT,pour reproduire l'expérience pour les MLM il faut utiliser la librairie torch. Nous avons pris le model "camembert". Dans la librairie il y a une méthode fillmask qui permet de lancer l'action de recherche et fait appel au modèle pour générer des résultats.
Dans le cas de Bert, il faut faire appel à la librairie transformers qui permettra d'appeler fillmask aussi.

Pour la partie fine tuning, il y a une succession de librairie à télécharger. Mais les plus importants sont tensorflow, tensorflowhub, Bert, numpy, tokenization et pandas.

Pour l'installation de Bert, il faut préparer tensorflowhub qui s'occupera de charger le modèle avec une URL. Cette URL doit être prise sur le site Tensorflowhub. Il contient une grosse majorité des modèles à jour en anglais.

Il faut bien entendu, gérer les dataset et les tokenizer avec la librairie tokenization.

Sachant que tensorflowhub contient que des modèles en anglais, il fallait chercher le modèle français dans une autre librairie, ici, on a choisi transformer, il suffit simplement de l'importer sans spécifier le modèle précis. De même que le cas précédent, il faudra la charger.

Il faut tokenizer de nouveau la dataset et effectuer du preprocessing.

Il est aussi important de noter, qu'il est recommandé d'effectuer toutes ces configurations sur anaconda. Il y a beaucoup de problèmes avec les versions des différentes librairies qui créent des conflit entre-elles.

Tout les modèles ainsi que la documentation sont disponible à l'adresse : https://github.com/ioudahya/Classification-context-eBERT-CamemBERT.git

IV. Résultats

A. Analyse MLM CamemBERT

Table I
Quelle est votre <mask> ?

Score	Résultat
0.08961806446313858	expérience
0.07944537699222565	spécialité
0.05311649292707443	profession
0.035945046693086624	devise
0.028544200584292412	personnalité

Nous pouvons remarquer que les résultats venant du pré-entraînement sont assez remarquable, avec une phrase simple contenant un mask, il trouve avec faciliter de bon résultat. Le score montre bien qu'il est bien sur des résultats donnés.

Essayons maintenant avec une phrase un peu plus compliquée à comprendre.

Table II
Hisser les voiles, <mask> !

Score	Résultat
0.09438372403383255	bonjour
0.04255270957946777	maintenant
0.03574129939079285	oui
0.030704302713274956	enfin
0.029116446152329445	voilà

Décidément, nous remarquons que le modèle n'est pas si parfait que ça. Il n'arrive pas à avoir des solutions correctes même si le score prétend le contraire. Ici le problème c'est qu'il est très instable pour comprendre le contexte et rend les phrases très étranges.

Table III
Rendez nous <mask> !

Score	Résultat
0.48574626445770264	visite
0.05029233545064926	demain
0.03767026960849762	compte
0.03340661898255348	nombreux
0.030758218839764595	célèbre

Nous pouvons voir ici, que lorsqu'il remplace le mask par visite, il n'est pas totalement sur du résultat, bien qu'il soit correct. Le reste des mots et des scores sont mauvais. Ce qui rend la tâche compliqué au modèle ici, ce sont les contextes de chaque phrase, surtout que la phrase est assez courte. Le modèle CamemBERT peut trouver une bonne solution pour la partie des phrase simple, mais rend la tâche beaucoup plus compliqué lorsque nous voulons exprimer un contexte à cette phrase.

B. Analyse MLM BERT

Table IV
The man worked as a [MASK].

Score	Résultat
0.04804684966802597	lawyer
0.037494439631700516	waiter
0.03551263362169266	cop
0.03127165883779526	detective
0.027423148974776268	doctor

Lorsque nous comparons les résultats de CamemBERT et celui de BERT on remarque que si la phrase est simple de cohérence, il trouve facilement des réponses avec un score qui est très sûr.

Table V
The best thing since sliced [MASK].

Score	Résultat
0.237198144197464	open
0.07578453421592712	off
0.05955592915415764	up
0.04394252970814705	bread
0.03836282715201378	meat

Nous essayons cette fois-ci avec une expression en anglais. Aucun contexte n'est mis en avant, mais on voit bien qu'il se complique la tâche. Bien qu'il ait trouvé la bonne expression, le reste des phrases sont étranges. Même si les données d'entraînement utilisées pour ces deux modèles peuvent être qualifiées de relativement neutres, les modèles peuvent avoir des prédictions biaisées.

C. Fine tune avec CamemBERT

Lors du fine tuning, nous avons entraîné les nouvelles couches de neurone et leur poids. Nous allons évaluer le modèle entraîné sur une plus petite dataset en nous basant sur le f1-score et la matrice de confusion.

Le f1-score est une métrique qui prend en compte la précision et le rappel sur une catégorie, ici 0 représente la satisfaction et 1 représente la déception. Le F1-score est compris entre 0 et 1, si le score est proche de 1, c'est que c'est le meilleur score possible.

Table VI
tableau de confusion

	precision	recall	f1-score	support
0	0.00	0.00	0.00	0
1	1.00	0.95	0.98	4000
accuracy	/	/	0.95	4000
macro avg	0.50	0.48	0.49	4000
weighted avg	1.00	0.95	0.98	4000

Nous obtenons un f1-score de : 0.98, c'est énorme ! Cela veut dire que notre modèle fonctionne très bien.

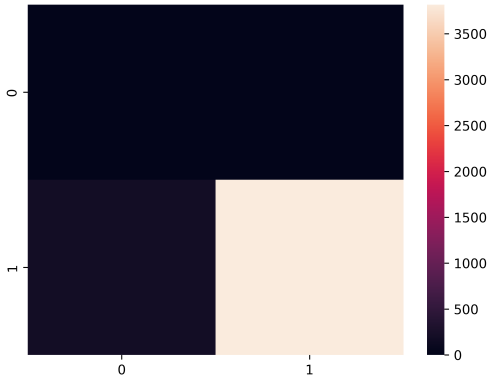

Figure 1. Voici la matrice de confusion qui est relié au tableau de confusion, nous permet de bien comprendre l'état de chaque prédiction.

Nous allons voir maintenant des exemples d'application.

Table VII
Phrase en français

phrase
Ce film est nul.
Ce film est trop cool.
Malgrer le bruit dans le cinéma, le film était bien.
Les films de denzel washington sont une -
tuerie ! les films sont toujours aussi bon.

Table VIII
Résultat en français

résultat
0.93138745680511
0.0044050012674175
0.9373445346138
0.0041305730548398

Pour la première solution, il est proche de 1, c'est qu'il considère que cette critique n'est pas satisfaite. Pour

le deuxième, il reconnaît bien que la critique est satisfaite. Pour le 3e, c'est un peu bizarre comme solution, car il pense que la critique n'est pas bonne, ce qui peut prêter à confusion, c'est le fait qu'il y ait du bruit dans la salle. Pour la dernière critique, il reconnaît qu'il a été satisfait.

D. Fine tune avec BERT

Sur un total de 100 millions de paramètres, avec le fine tune , nous avons ajouté 9,500 millions de paramètres en plus.

Pour cette partie, nous allons directement appliquer les différentes prédictions sur le modèle.

TABLE IX
PHRASE EN ANGLAIS

phrase
I liked this movie
Everything was fine but not the movie !
Wow ! this movie was so good but i didn't like the ending.
Denzel Washington is such a good actor, -
I love the way he act in this movie

TABLE X
RÉSULTAT EN FRANÇAIS

résultat
0.9682432
0.16217496
0.1922773
0.9889747

Ici, quand le résultat est proche de 1, c'est que la critique est positive sinon négative. Pour la première phrase, nous pouvons voir qu'il est satisfait, pour la deuxième, il reconnaît bien que le film n'était pas bon malgré la tournure de la phrase. Pour la troisième, il reconnaît qu'il n'a pas aimé la fin du film. Et pour le dernier, il reconnaît bien qu'il a aimé le film malgré le fait qu'il parle de l'acteur. Pour ce modèle, nous pouvons donc comprendre que l'analyse du contexte n'est pas un problème. Il comprend tout à fait les tournures des phrases.

Nous avons évalué BERT en lui introduisant toute la dataset servant d'entraînement et nous avons eu deux valeurs de retour. 0.29 de loss et 0.86 en prédiction. Ce qui veut dire que durant une grande majorité de prédiction, il trouvera le bon résultat.

E. Comparaison des deux modèles

Nous pouvons remarquer que lorsque nous avons analysé les deux modèles, les prédictions chez CamemBERT ont une difficulté à comprendre le contexte, mais BERT réussi malgré les tournures de phrase. Ce qu'il faut comprendre ici, c'est que la langue française est compliquée à comprendre. Les tournures jouent souvent sur quelques pronoms et laissent le contexte choisi par CamemBERT à désirer. Même si CamemBERT à un bon score lors de son évaluation, il arrive quand même à avoir de mauvais résultat. Du côté de BERT, même si son évaluation n'est pas aussi surprenante que celle de CamemBERT, nous remarquons qu'il est plus performant. N'oublions pas que tout ceci résulte, du changement des hyperparamètres et des dataset données au modèle. Ici, étant limité par les composants de la carte graphique, nous avons pris de grande dataset et de petit hyperparamètres.

V. DISCUSSION

En partant des observations faites dans la section "Résultats", nous comprenons mieux les modèles BERT et CamemBERT, leur analyse du langage est certes similaire, mais lors de l'entraînement tout change. La langue française étant plus complexe dans son analyse, nous avons du mal à reconnaître quelques contextes précis. Comme dans la tournure de la phrase où l'ambiance de la salle était mauvaise, mais que le film était bon (cf TABLE VII).

Bien que les modèles ont une bonne évaluation, tout peut changer en modifiant la dataset. Nous avons obtenu un f1-score de 0.98, mais les résultats n'étaient pas aussi bien que celui de BERT, qui a eu un résultat de 0.86.

Il faut bien faire attention au hyperparamètre donnée, tout peut changer, si on modifie la fonction d'activation softmax avec une autre formule, le résultat ne sera pas le même. De même pour l'algorithme d'optimisation. Mais, malgré le fait que le modèle aient quelques erreurs dans les résultats, les modèles restent tout de même puissant. Il a suffi de rajouter quelques millions de paramètres pour avoir une aussi grande précision. Qu'en est-il du changement des epochs ? du batch-size ? des base de données ? Nous n'avons pas tout exploré, mais il serait bien entendu intéressant de comparer les différents modèles avec différents paramètres. Ceci permettre de trouver la meilleure configuration pour ce type d'entraînement.

VI. CONCLUSION

Nous avons appris des choses importantes à propos des modèles utilisant le NLP. Pour commencer, nous savons qu'il est possible de fine tuner les modèles à un plus haut niveau que le pré-entraînement. Le français requiert beaucoup plus d'entraînement que l'anglais puisque la langue est un peu plus difficile.
Il est important de bien choisir les hyperparamètres et les bonnes dataset à joindre avec le modèle utilisé. Tout ceci doit bien entendu être accompagné d'un meilleur matériel afin de mieux entraîner.

Pour conclure, nous constatons l'importance de la conception des modèles comprenant le NLP. Comme expliqué dans l'introduction, cette compréhension impacte fortement l'analyse des données du partage de l'information sur Internet.

Références

[1] Pedro Javier Ortiz Suárez Yoan Dupont Laurent Romary Eric Villemonte de la Clergerie Benoît Sagot Djamé Seddah Louis Martin, Benjamin Muller. Les modèles de langue contextuels camembert pour le français : impact de la taille et de l'hétérogénéité des données d'entrainement. *JEP-TALN-RECITAL*, 220.

[2] naman jingfeidu danqi omerlevy mikelewis lsz ves yinhanliu, myleott. Roberta : A robustly optimized bert pretraining approach. *Facebook AI*, 2019.

[3] Kenton Lee Kristina Toutanova Jacob Devlin, Ming-Wei Chang. Bert : Pre-training of deep bidirectional transformers for language understanding. *Google AI Language*, 2019.

[4] laurent romary eric de la clergerie djame seddah benoit sagot benjamin muller, pedro ortiz. Camembert : a tasty french language model. *Association for Computational Linguistics*, 2020.