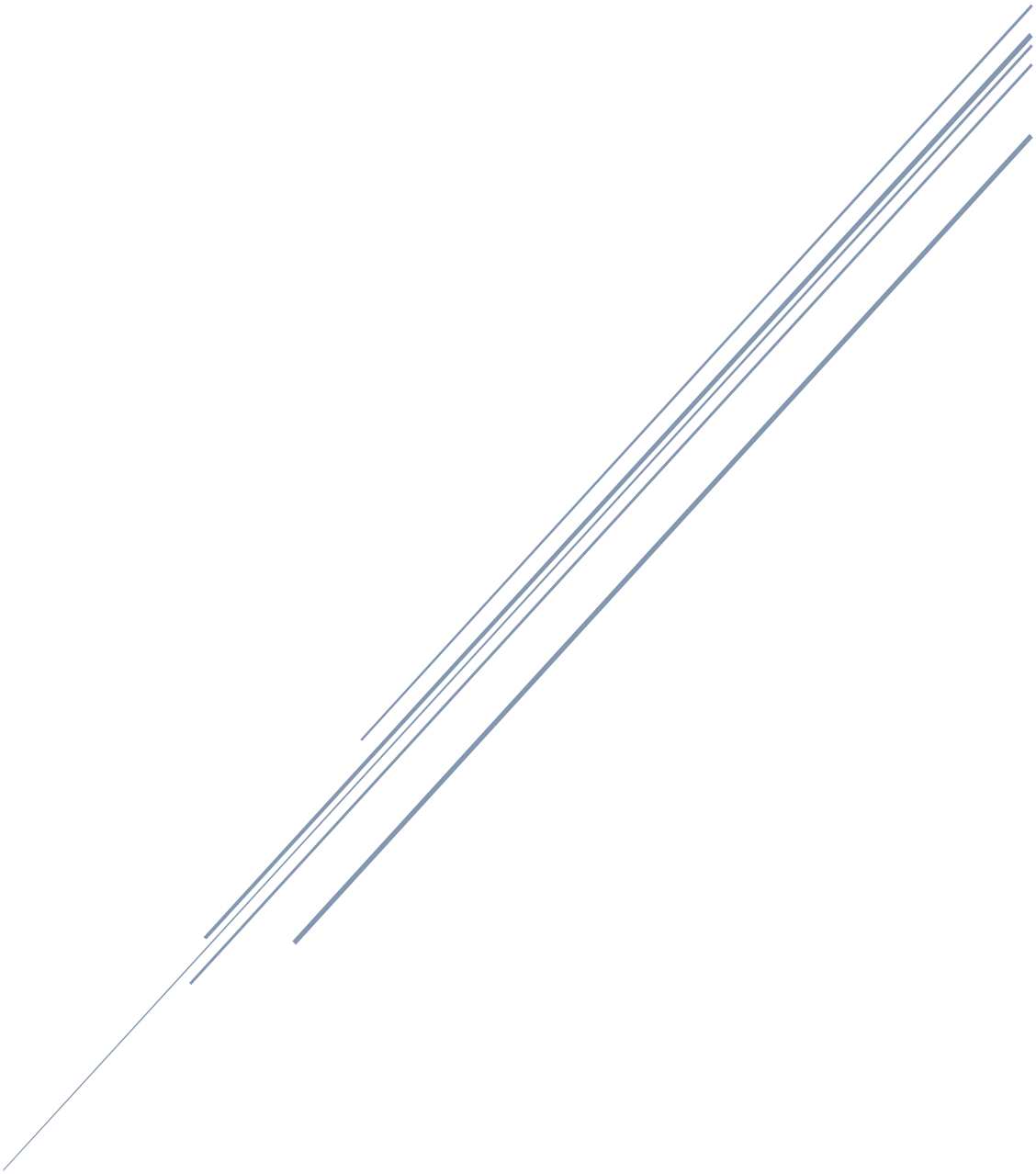
DEBERTAV3

Analyse de sentiment de TweetEval Emotion



Thibaud GROSJEAN

OpenClassrooms – Ingénieur Machine Learning – Mai 2022

SOMMAIRE

Introduction *3*

Présentation du modèle *3*

Mecanisme d'Attention **3**

Architecture **4**

Pré-Entrainement **5**

Partage des Embeddings **6**

Descente de Gradients **6**

Expérimentation *7*

Contexte **7**

Données **7**

Implémentation **8**

Resultats **9**

Conclusion *9*

Références *9*

INTRODUCTION

Le modèle BERT (Bidirectional Encoder Representations from Transformers) [1] est l'un des premiers modèles transformeur (transformer). BERT est un modèle auto-attentif, ce qui lui permet de développer une compréhension statistique du langage avec lequel il a été entrainé sans requérir d'étiquetage des données (data labelling). Autrement dit, ce type de modèle peut être entrainé de manière auto-supervisée (self-supervised learning). C'est un avantage considérable dans le domaine du traitement du langage naturel (Natural Langage Processing - NLP) où la création de jeux de données supervisées représente un travail considérable pour l'homme.

Le mécanisme d'auto-attention à têtes multiples (multi-head attention) implémenté dans BERT a également grandement amélioré l'efficience de l'entrainement comparativement aux algorithmes de pointe précédents et règle notamment la problématique des réseaux de neurones de type RNN (Recurent Neural Networks) et CNN (Convolutional Neural Networks) qui peinent à encoder les vecteurs de contexte des séquences (embeddings) à dimensionalité élevée.

Le modèle BERT peut aisément traiter des séquences d'une longueur maximale de 512 éléments (tokens) tout en étant hautement parallélisable. Cet avantage a permis de pré-entrainer (pre-tuning) le modèle sur de larges quantités de données dans des tâches de Masked Langage Modeling (MLM) et de Next Sentence Prediction (NSP), il a ainsi acquis une compréhension de la langue anglaise. Il peut être spécialisé (fine-tuning) dans de nombreuses tâches de NLP à l'aide d'un unique GPU et offre d'impressionnantes capacités de généralisation.

De nombreuses itérations du modèle BERT ont été développées dans le but d'améliorer ses performances et son efficience. Nous avons spécialisé un modèle de pointe (state-of-the-art), variante de BERT : le modèle DeBERTaV3 (base) [2] dans le but de détecter 4 types d'émotion dans les tweets du jeu de données (dataset) TweetEval Emotion [3].

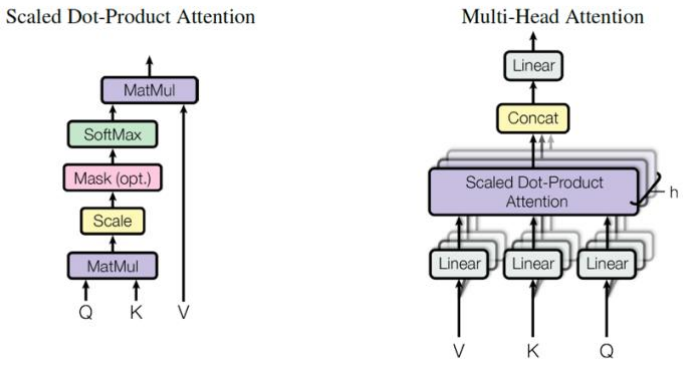
DeBERTaV3 apporte des innovations fonctionnelles au modèle BERT : il améliore le mécanisme d'auto-attention grâce à l'attention démêlée (distangled attention), sa descente de gradients (gradient descent), et intègre l'amélioration du pré-entrainement (pre-tuning) avec la tâche de détection des tokens remplacés (Replaced Token Detection - RTD) implémentée dans ELECTRA [4].

PRESENTATION DU MODELE

MECANISME D'ATTENTION

Contrairement aux anciens modèles transductifs de type RNN et CNN, le transformeur se passe complètement des mécanismes de récurrence et de convolution pour générer des dépendances entre les données d'entrée (inputs) et les cibles (outputs). BERT crée des représentations en se basant uniquement sur le mécanisme d'attention ce qui lui permet de se défaire en partie de la nature séquentielle (accumulation d'états profonds - hidden states) des anciens modèles, d'augmenter la taille des batchs et de favoriser la parallélisation. Avant l'apparition des transformeurs, le mécanisme d'attention était implémenté dans les encodeurs-décodeurs à base de RNN sous les formes proposées par Bahdanau et al. (2014) et Luong et al. (2015).

Le mécanisme d'auto-attention de BERT (et de ses itérations) capture les relations entre les différents termes (mots) de la séquence d'entrée. Il est proposé sous la forme d'un produit scalaire (scaled-dot product attention) des paires clés-valeurs, ce qui permet d'obtenir une matrice dont le calcul peut être grandement optimisé comparativement à l'attention additive (additive attention). Vaswani et al. (2017) ont construit le mécanisme d'attention multiple (multihead attention) à partir de l'attention scalaire ce qui permet à la fonction d'attention d'extraire des informations de différents sous-espaces de représentation (representation subspaces) à des positions différentes de manière parallèle et simultanée.

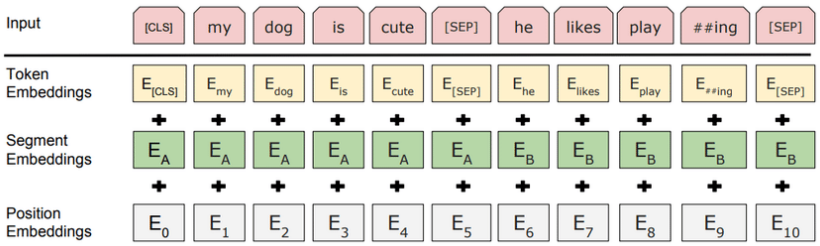


Attention scalaire (à gauche) et attention multiple (à droite)

DeBERTaV3 apporte une amélioration de la l'attention scalaire multiple en proposant l'attention démêlée qui permet d'intégrer la représentation de la position relative des termes de la séquence grâce à la génération d'un vecteur supplémentaire. La distance relative entre les termes d'une séquence est d'une importante cruciale dans la représentation du sens des mots selon Pengcheng He et al. (2021). Cette solution consiste à calculer 4 scores : *contenu vs contenu*, *contenu vs position*, *position vs contenu* et *position vs position* traduit les positions relatives des termes et permet de résoudre cette problématique.

ARCHITECTURE

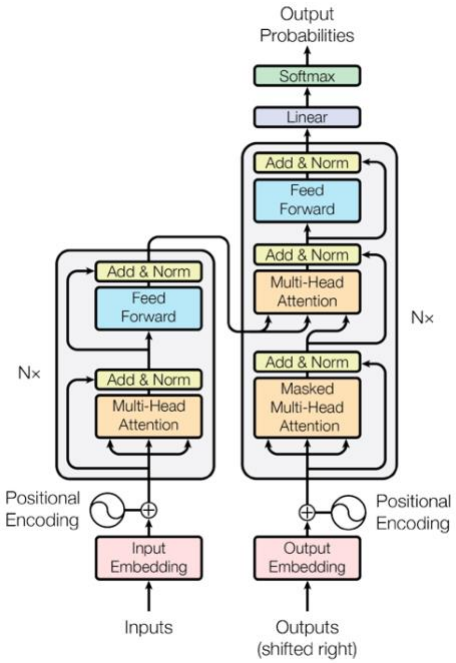
Le transformeur est composé de deux modules : le premier, l'encodeur, encode les données d'entrée sous la forme de de texte brut après une étape de tokenisation, ce qui permet d'obtenir les séquences : des vecteurs numériques aux valeurs continues qui représentent les caractéristiques des données d'entrée afin d'être ingérées par le modèle.



Exemple de séquences des Transformeurs (BERT)

On retrouve l'attention multiple sous diverses formes dans les modules du transformeur. Sous sa forme la plus classique, qui mimique le mécanisme présent dans les modèles sequence-to-sequence (vec2vec), les têtes d'auto-attention ont accès à l'ensemble des termes de la séquence d'entrée de manière instantanée, c'est le cas pour les couches (layers) de d'attention multiple présents dans l'encodeur et pour les couches encodeur-décodeur du décodeur.

Différemment, les couches d'attention multiple du deuxième module, le décodeur, décodent les séquences de manière itérative de la même façon qu'un humain lirait une phrase : en commençant par le début de la séquence, puis à chaque itération en intégrant le terme suivant et ce jusqu'à ce que la séquence soit traitée entièrement. A chaque itération, le mécanisme d'attention permet au modèle de construire, pour chaque terme, des vecteurs d'attention qui sont intégrés à l'itération suivante constituant ainsi un modèle autorégressif.



Architecture du modèle transformeur

PRE-ENTRAINEMENT

Le pré-entrainement du transformeur est l'étape d'entrainement durant laquelle le modèle crée les représentations du langage (séquences) sur lequel il est entrainé avant de pouvoir être spécialisé sur d'autres tâches. Cette phase, effectuée sur de larges quantités de données de manière auto-supervisée en comparant l'utilisation des mots dans des contextes différents requiert une puissance de calcul GPU massive. DeBERTaV3 a été pré-entrainé sur un jeu de données représentant un total de 161GB.

Model	Wiki+Book 16GB	OpenWebText 38GB	Stories 31GB	CC-News 76GB	Giga5 16GB	ClueWeb 19GB	Common Crawl 110GB	CC100 2.5TB
BERT	✓							
XLNet	✓				✓	✓	✓	
RoBERTa	✓	✓	✓	✓				
DeBERTa	✓	✓	✓					
$\text{DeBERTa}_{1.5B}$	✓	✓	✓	✓				
DeBERTaV3	✓	✓	✓	✓				
mDeBERTa_{base}								✓

Comparaison des jeux de données d'entrainement des transformeurs

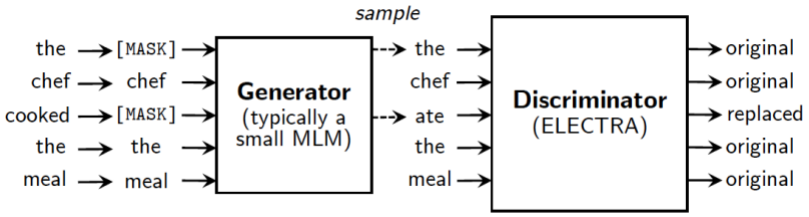
Le modèle BERT a été entrainé sur les deux taches que sont la MLM et la NSP, mais c'est grâce à la tâche de MLM que le modèle crée ses représentations. Nombre de transformeurs de pointe se passent de la tâche de NSP, c'est le cas pour ELECTRA et DeBERTaV3 car de récents travaux suggèrent que cette tâche n'améliore pas les performances du modèle (Yang et al., 2019; Liu et al., 2019).

Dans le cadre de la tâche de MLM de BERT, l'encodeur masque aléatoirement 15% des termes générés pour les remplacer par un terme spécial (« [MASK] », typiquement). L'objectif du décodeur est de déterminer quels vrais termes se cachent derrière les termes masqués (corrompus), c'est de cette manière que le modèle apprend à représenter le langage. Cette approche est utilisée par DeBERTa jusque dans sa version 2.

L'approche classique de la MLM présente un inconvénient : le modèle ne peut apprendre que de ces 15% de termes masqués (Kevin Clark et al., 2020). Avec ELECTRA, les masques de la MLM sont remplacés par des termes

synthétiques mais néanmoins plausibles. Durant la phase de pré-entrainement, le décodeur est remplacé par un discriminateur dont la tâche est de détecter les termes corrompus.

Cette approche de détection des termes remplacés (RTD) dans le style des Generative Adversarial Networks (GANs) force le modèle à scruter l'ensemble des termes de la séquence, ce qui améliore grandement l'efficience de l'entrainement. Une fois le pré-entrainement finalisé, le discriminateur est supprimé pour être remplacé par un décodeur classique initialisé avec les poids du discriminateur. DeBERTaV3 emprunte cette stratégie de pré-entrainement à ELECTRA afin d'améliorer ses performances.



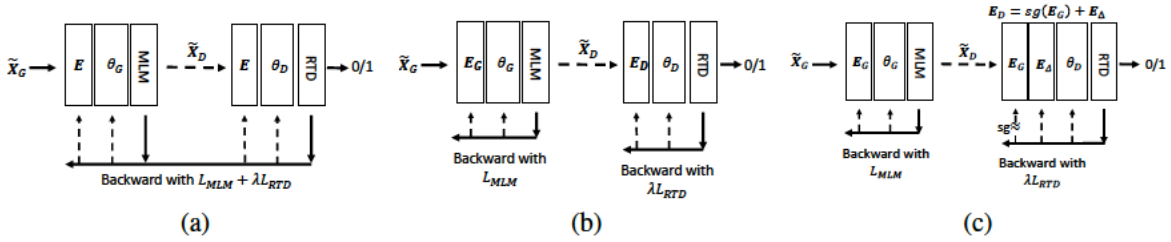
Fonctionnement du discriminateur d'ELECTRA (et de DeBERTaV3)

Partage des Sequences

Durant le pré-entrainement d'ELECTRA, les séquences sont partagées entre le générateur et le discriminateur (Embedding Sharing - ES). Pour chaque passe en avant (forward pass) du modèle, la perte est calculée par combinaison de la perte du générateur et du discriminateur durant la rétropropagation du gradient (back-propagation). Le fait que cette perte soit partagée provoque une situation de tir à la corde (tug-of-war) entre les mises à jour des séquences requises par chacun des deux modèles.

Selon Pengcheng He et al. (2021), cette solution n'est pas optimale car les objectifs de MLM et de RTP étant drastiquement différents, la mise à jour des séquences peut se faire de façon extrême et opposée afin de favoriser l'une des deux tâches. Il faut alors grandement contrôler la vitesse d'apprentissage (learning rate) pour obtenir la convergence du transformeur vers l'optimum.

Dans une configuration de non-partage des séquences (No Embedding Sharing - NES), les séquences du générateur et du discriminateur sont différents et mis à jour de manière alternée. Les auteurs démontrent que cette stratégie est plus efficiente et permet d'obtenir une convergence plus rapide. En revanche, la NES n'offre pas des performances plus avantageuses que l'ES.



(a) ES, (b) NES, (c) GDES

Descente de Gradients

Afin des conserver à la fois les avantages de l'ES qui présente des performances plus élevées que la NES, et l'efficience de la NES, Pengcheng He et al. (2021) proposent le partage de gradient des séquences démêlées (Gradient-Disentangled Embedding Sharing - GDES), une approche qui consiste à restreindre la descente de gradients pendant l'entraînement du modèle.

Avec GDES, les gradients du RTD sont calculés uniquement sur la base de la perte du MLM ce qui permet d'éviter d'être dans une configuration de tir à la corde. Dans cette configuration, les gradients du MLM sont utilisés pour actualiser à la fois les séquences du générateur et du discriminateur alors que les gradients du discriminateur servent exclusivement à actualiser les séquences du discriminateur.

EXPERIMENTATION

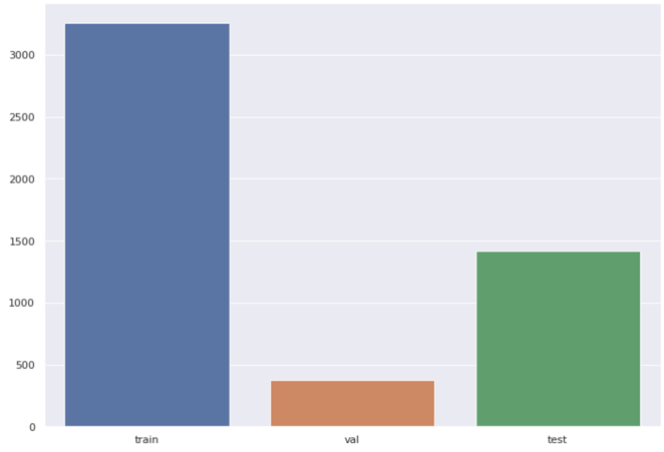
CONTEXTE

NowUrCrown est une startup qui développe une plateforme de suivi d'audience pour les influenceurs du célèbre réseau social Twitter. Avec son application, elle propose notamment un tableau de bord présentant des statistiques et leurs visualisations et souhaite augmenter son offre de services (et d'indicateurs) grâce à des modèles prédictifs.

L'un des objectifs de l'entreprise est d'avoir la capacité de détecter les sentiments présents dans les tweets afin d'offrir aux influenceurs une compréhension plus poussée de leur audience. NowUrCrown a d'ores et déjà implémenté l'algorithme Universal Sentence Encoder (USE) dont les performances ne sont pas satisfaisantes, elle a donc fait appel à nous pour implémenter un modèle de pointe dans une preuve de concept (POC).

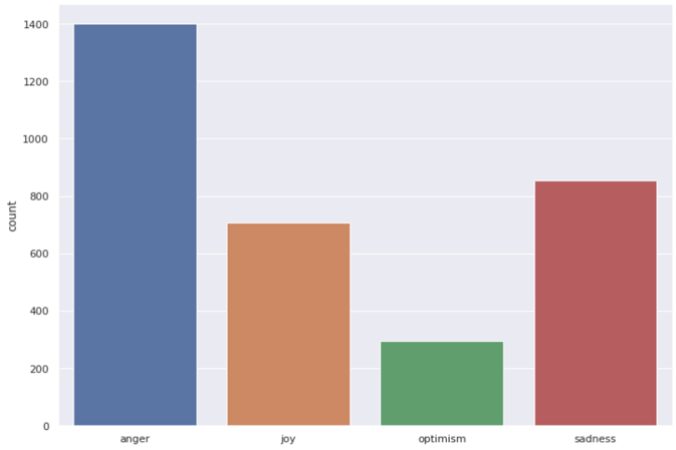
DONNEES

Nous avons utilisé le jeu de données TweetEval afin de réaliser le POC. Le jeu de données est composé de 7 (sous) jeux de données qui permettent d'entrainer des algorithmes sur des tâches de classification binaire et multiple. Nous avons sélectionné la tâche « Emotion », qui regroupe 4 classes : « colère » (« anger »), « joie » (« joy »), « tristesse » (« sadness »), « optimisme » (« optimism »), et avons donc fait face à une problématique de classification multiple.



Distributions des populations par jeu de données

Le jeu de données TweetEval Emotion regroupe 5052 tweets, dont 3257 dans son jeu d'entraînement, 374 dans son jeu de validation et 1421 dans son jeu de test. Les classes présentes dans le jeu d'entrainement sont déséquilibrées, la classe modale « anger » représentant 43% de l'ensemble des échantillons. Le jeu de validation présente une distribution similaire.



Distribution des classes dans le jeu d'entrainement

La métrique de référence pour la tâche Emotion est le score F1 macro-moyenné, un score adapté à la classification déséquilibrée.

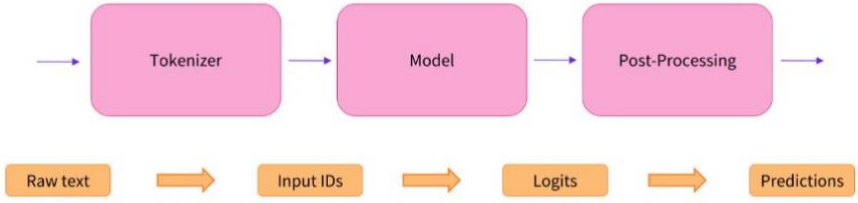
IMPLEMENTATION

Nous avons implémenté une méthode naïve (classe modale - CM) ainsi que 3 algorithmes classiques (Régression Logistique - RL, USE, et BERT (base)) afin de constituer une baseline. Nous avons ensuite comparé les résultats avec notre modèle de pointe : DeBERTaV3 (base). La méthode naïve CM a consisté à prédire uniquement la classe modale afin d'obtenir un score de base.

Les algorithmes RL et USE ont requis un pré-traitement (preprocessing) des données avant de pouvoir être entrainés. Nous avons créé un pipeline afin de supprimer les caractères spéciaux, les stopwords (mots redondants), appliqué une racinisation (stemming), et transformé les données à l'aide d'une matrice inverse de document (TFIDF, uniquement pour la RL). Ces transformations (exceptée la dernière) ont également permis d'explorer et de visualiser les caractéristiques des données.

Pour les transformeurs (BERT et DeBERTaV3) le tokenizer applique l'ensemble des étapes de pré-traitement directement sur des données non traitées (texte brut) : le tokenizer applique une tokenisation (tokenization) end-to-end à l'aide de modèles tels que la tokenisation Wordpiece. Le tokenizer permet également d'inverser les logits (outputs) du modèle afin d'obtenir des prédictions lisibles par l'homme.

Dans leur configuration de base, ces modèles disposent de 6 couches d'attention dans leur encodeur et de 6 dans leur décodeur. En sortie, le décodeur dispose d'une couche linéaire de 768 neurones sur laquelle nous avons apposé une couche linéaire de pré-classification de 768 neurones suivie d'une couche de drop-out (0.3) et d'une couche linéaire de classification de 4 neurones (correspondant au nombre de classes). Nous avons ré-entrainé l'ensemble des couches de ces modèle sur 3 epochs avec une vitesse d'apprentissage de 1e-5.



Pipeline de spécialisation et d'inférence des transformeurs

RESULTATS

Sur le jeu de validation, l'algorithme DeBERTaV3 dépasse largement, sans surprise, notre algorithme baseline de référence USE avec un score F1 macro moyenné qui s'élève à 76.8. Cette performance est acceptable et démontre que la tâche peut être accomplie par un modèle de pointe. Nous avons relevé que le jeu d'entrainement de TweetEval Emotion est plus difficile à prédire que son jeu de test.

Validation	CM	LR	USE	BERT (base)	DeBERTaV3 (base)
F1 macro	14.9	30.5	45.1	71.4	**76.8**
Accuracy	42.7	43.8	60.6	77.5	**83.6**

Performances sur le jeu de validation

Nos expérimentations démontrent que le modèle DeBERTaV3 offre des performances de pointe sur le jeu de test. DeBERTaV3 dépasse le score (F1 macro-moyenné) de BERT de 5.2 points et le score du modèle TimeLMs [5], présent en tête du classement officiel de TweetEval Emotion, de 2.2 points.

Test	CM	LR	USE	BERT (base)	TimeLMS	DeBERTaV3 (base)
F1 macro	14.0	28.2	54.4	77.2	80.2	**82.4**
Accuracy	39.2	41.8	65.1	80.5	-	**84.7**

Performances sur le jeu de test

La performance de DeBERTav3 s'explique en partie par le fait que le modèle arrive à mieux détecter la classe minoritaire « optimisme », dont la population représente seulement 9% du jeu d'entrainement (294 des 3257 échantillons). Avec DeBERTaV3, le score F1 de cette classe s'élève à 74, contre 62 pour BERT, il n'est que de 18 avec USE. Le modèle présente une meilleure capacité à généraliser que BERT et semble capturer les caractéristiques de chaque classe avec plus de finesse.

CONCLUSION

Afin de démontrer comment résoudre la problématique de NowUrCrown, nous avons implémenté un modèle de pointe, DeBERTaV3, et comparé ses performances à une baseline. Nous avons démontré que DeBERTaV3 dépasse largement le score F1 macro-moyenné de notre algorithme de référence USE sur le jeu de validation ainsi que le score de BERT (de 5.2 points) et de TimeLMs (de 2.2 points) sur le jeu de test de TweetEval Emotion.

RÉFÉRENCES

[1] Ashish Vaswani et al, Attention Is All You Need, 2017

[2] Pengcheng He et al., DeBERTaV3: Improving DeBERTa using ELECTRA-Style Pre-Training with Gradient-Disentangled Embedding Sharing, 2021

[3] Francesco Barbieri et al., TWEETEVAL: Unified Benchmark and Comparative Evaluation for Tweet Classification, 2020

[4] Kevin Clark et al., ELECTRA: Pre-Training Text Encoders Aa Discriminators Rather Than Generators, 2020

[5] Daniel Loureiro et al., TimeLMs: Diachronic Language Models from Twitter, 2022