CANINE: Pre-training an Efficient Tokenization-Free Encoder for Language Representation

Group N4-26 – Project Report on Topic 4

Anis Oueslati
Ecole Normale Supérieure
oueslati.anis83@gmail.com

Matias Etcheverry
Ecole Normale Supérieure
matias.etcheverry9@gmail.com

Zeki Topçu
Ecole Normale Supérieure
zekitopccu@gmail.com

Abstract

In this project, we investigated the performance of a state-of-the-art natural language processing model, CANINE. We sought to fine-tune the model on a question answering task and compare it against a multi-lingual model on a classification task.

1. Introduction

Natural language processing pipeline are usually segmented in multiple block, one of which is the tokenizer. Most of the time, this block allows to separate words into consistent subwords, in order to reduce the size of the vocabulary. However, such approach raises some issues. It is particularly suitable for Latin languages, but very unstable for character languages like Mandarin or Arabic. To overcome this problem, [2] has published a character based tokenizer. In addition, this tokenize has other advantages, such as being robust to spelling mistakes. Our code and results are available on GitHub [1]. The models were implemented on Google Colab and on Kaggle, These 2 platforms provided us with free GPUs allowing us to run computationally intensive trainings. However, the available memory is limited, as well as the existence of additional constraints, such as execution time limits (disconnection after a certain time), or the absence of free GPUs.

2. CANINE

CANINE, from Character Architecture with No tokenization In Neural Encoders, is a transformers model designed for natural language understanding tasks. It is pretrained on a large corpus multilingual data in a self-supervised fashion, like BERT. Thus, CANINE does not use human labeled data during pretraining, instead it uses solely raw text.

CANINE uses a character-based input representation, which means it processes text at the character level rather than the word or subword level. CANINE's architecture consists of the following components:

- Character Embedding: Each character in the input sequence is converted into a high-dimensional vector representation, capturing the semantic information of the character.
- Transformer Layers: These layers process the character embeddings through self-attention mechanisms and feed-forward networks, enabling the model to learn contextual relationships between characters and generate meaningful representations of the input text. Note that it uses no less than 3 Transformer encoders internally: 2 "shallow" encoders (which only consist of a single layer) and 1 "deep" encoder (regular BERT encoder).
- Pooling Layer: The output of the transformer layers is pooled to generate a fixed-size representation of the input sequence, which is used for downstream tasks such as classification or translation.
- Task-specific Layers: Depending on the specific task, additional layers may be added on top of the pooled representation to generate the final output, such as a dense layer for classification or a decoder for translation.

Differences from mBERT mBERT [3], or multi-language BERT, is a language model trained on text from multiple languages. It is a commonly used tool for natural language processing tasks, like translation, text classification, and question-answering. A few differences arise between CANINE and mBERT. CANINE uses a character-based input representation, while mBERT works with word pieces, which are smaller units of a word. This makes CANINE better at handling rare and unknown words or languages with complex structures, like Arabic or Chinese. Apart from the representation of inputs, they don't share the same model structure. Although both models are based on the Transformer architecture, there are differences in their specific configurations, such as the number of layers, attention heads, and other hyperparameters. These differences can lead to variations in the models' learning capabilities and performance on various tasks.

In summary, CANINE and mBERT are both neural network models designed for natural language understanding tasks, but they differ in their input representation and model

[1] https://github.com/MatiasEtcheve/SLP-CANINE

structure. CANINE works with characters, while mBERT uses word pieces, which impacts their ability to handle different languages and text structures.

3. Fine-tuning the CANINE Model on Tydi QA Dataset

The first objective of our project is to fine-tune a CANINE model on a question answering task, in order to reproduce the results proposed by the authors. To do so, we choose to fine-tune on the Tydi QA dataset.

3.1. The TYpologically DIverse languages with Question-Answer dataset

TyDi QA [1] is a multilingual question answering dataset designed to evaluate models' ability to answer questions in diverse languages and scripts. It contains over 200k question-answer pairs in 11 languages (English, Arabic, Bengali, Finnish, Indonesian, Korean, Russian, Swahili, Telugu, Turkish, Vietnamese). For each question, a full article from Wikipedia is used to extract the correct answer. This dataset is specifically adapted to our problem as there is a huge variety of language. The questions in the dataset are written by native speakers and cover a a wide range of topics, like history, science, entertainment etc.

We provide examples of questions in the dataset, in English and in Arabic. Some of the shortest questions are: Where is Belize? and ما هو العث؟ (What are dust mites?). The item longest question is كم عدد الدول الموقعة على اتفاقية العهد الدولي الخاص بالحقوق الاقتصادية والاجتماعية والثقافية؟ (How many countries are signatories to the International Covenant on Economic, Social and Cultural Rights?). The longest questions are not necessarily related to the shortest answers. Two tasks arise from the dataset:

- the *Minimal Answer task*: given a Wikipedia article and a question on this article, the goal is to find the start index and end index of a passage answering the question in the article. The passage should be the smallest possible. Multiple annotations can be available for this task. Typical metrics for this tasks are the distances to start and end indices, as well as classification metrics.

- the *Passage task*: given a Wikipedia article and a question on this article, the objective is to find the paragraph index answering the most the question. Typical metrics for this tasks are classification metrics like F1-score.

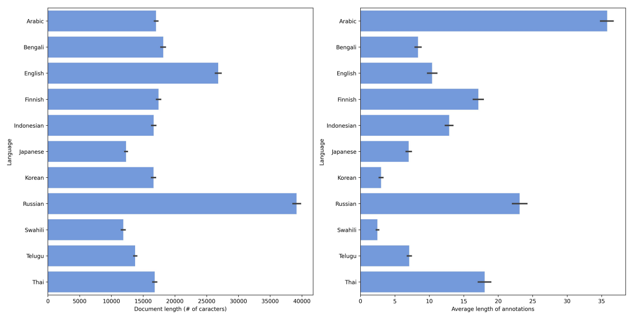

Figure 1. Length of documents and length of annotations in the original *Tydi QA* set

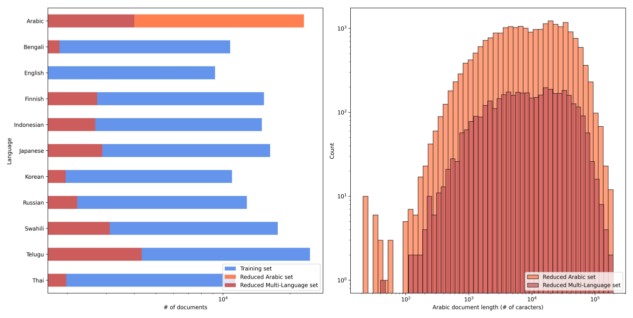

Figure 2. Number of documents and length of Arabic document in each split of the *Tydi QA* dataset

fig. 1 shows the distributions of lengths of articles and annotations per language. These annotations only focus on the *Minimal Answer task*. A short annotation means that the question can be answered in a few words while longer annotations mean that multiple paragraph are required to answer the question. We notice that the languages don't have the same distribution of length in document and in annotations. While Russian usually have longer articles, Arabic provides longer answers.

The dataset is initially made of 200 000 training reviews. Only tokenizing the dataset would require 12h of computation non stop. That is why, we decided to reduce the datatset size. We thus create 2 datasets:

- one Arabic only dataset made of 23 092 articles
- one multi language dataset made of 29 337 articles

fig. 2 shows the number of articles dedicated to each dataset. Creating smaller dataset will also benefit the training time.

3.2. Finetuning CANINE

As the fine-tuning process requires extensive computing resources, we chose to fix some of the hyperparameters of the CANINE model to ensure a reasonable training time. Specifically, we set the training batch size to 4 and the maximum sequence length to 2048. These values were chosen based on a trade-off between computational efficiency and model performance. Moreover, setting the context window to 2048 in CANINE roughly corresponds to setting the maximum sequence length to 512 tokens in usual language models (*e.g.* BERT)

3.2.1 Comparing learning capabilities between character based loss and subword based loss

In this part of the project, we aimed to compare the learning capabilities of the CANINE model when trained with two different types of loss: character-based and subword-based. The objective was to determine which type of loss would yield better performance when fine-tuning the CANINE model on the Tydi QA dataset. The training curves for both types of loss were plotted, as shown in Figure 3. The results showed that both types of loss had very similar behavior, indicating that the performance of the model was not significantly affected by the type of loss used. Therefore, we decided to focus on the character-based model for the rest of the project.

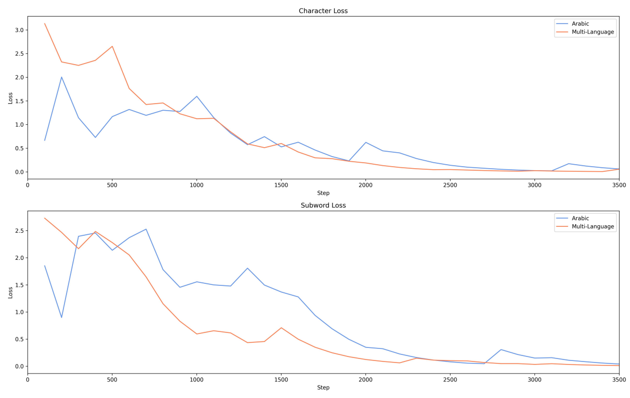

Figure 3. Training curves for the 2 types of loss on the *Tydi QA* dataset

3.2.2 Training for longer

In this experiment, our objective is to train the models for a longer period and see if the results align with those reported in the original paper. We use two different models: one trained on Arabic only, and another trained on multiple languages.

We evaluate the models using various metrics, loss and distance to start index and distance to end index, as shown in fig. 4. The models show significant improvement in the beginning, followed by slower progress towards the end of the training. However, they continue to improve over time, albeit slightly.

Another set of metrics we use for evaluation are precision, recall, and F1-score for the two tasks, as showed in Tab. 1. Our fine-tuned model performs better than the one proposed by the authors. It is important to note that the training and test sets are not the same, which could lead to different results.

We find it surprising that a model fine-tuned specifically on Arabic does not perform much better than a model fine-tuned on multiple languages when tested on an Arabic-only dataset. This observation suggests that the multi-language model might have effectively captured the nuances of the Arabic language during training, making it a strong competitor to the Arabic-only model.

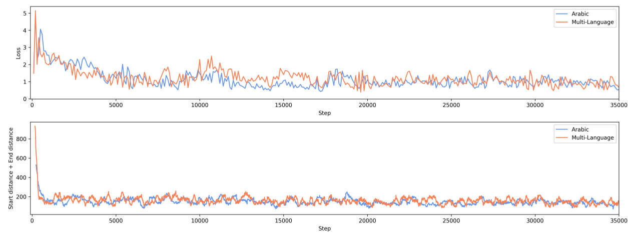

Figure 4. Training curves for the 2 models on the 2 datasets

Model	Fine-tuned on	Tested on	F1-score Minimal Answer Task	F1-score Passage Selection Task
Original	Multi Language	Multi Language	0.530	0.657
Ours	Arabic	Arabic	**0.9103**	**0.7085**
Ours	Multi Language	Arabic	0.8874	0.6863

Table 1. Test metrics from training reproducing results from CANINE

4. Finetuning CANINE on a classification task

The second experiment of our project is to use CANINE on a classification task and test it against mBERT.

4.1. The Multilingual Amazon Reviews Corpus dataset

The *Multilingual Amazon Reviews Corpus* [4] is a dataset comprising product reviews from the Amazon website, available in six languages: English, French, Spanish, German, Japanese, and Chinese. Each record in the dataset contains the review text, review title, star rating, anonymous reviewer ID, anonymous product ID, and a coarse-grained product category, such as books, gadgets, or clothes.

The dataset consists of 3 600 000 reviews, evenly divided into three splits. To reduce computational resources and decrease the time required for one epoch, we reduce the size of the training set to 30,000 reviews and the validation set to 1,000 reviews.

The distribution of length has variations across languages and categories, as shown in fig. 5. Document and title lengths also differ across languages. The categories are not evenly distributed, although the distribution of star ratings remains consistent across them, as illustrated in fig. 6.

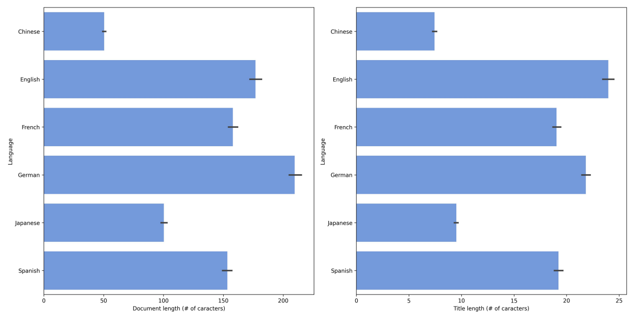

Figure 5. Length of documents and titles in the *Amazon Review Corpus* dataset

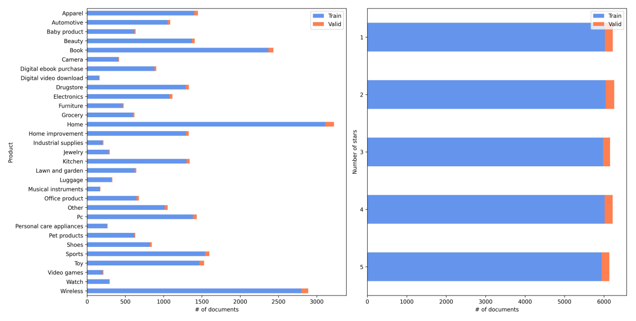

Figure 6. Category and stars distribution in the *Amazon Review Corpus* dataset

Predicting the number of stars is challenging due to the subjectivity of rating scales, making it impossible to achieve hundred percent accuracy. The length of reviews does not necessarily correlate with getting high recommendation. For instance, some of the shortest reviews include a one-star review saying ”I never received it,” and a two-star review titled ”Coucou” with the description ”Usuel et sport Déçu.” In contrast, one of the longest reviews, titled ”Who are all these people leaving 5-star reviews?” has a three-star rating and begins with a detailed account of the reviewer’s size and clothing preferences before discussing the product further.

4.2. Comparing CANINE and a multi Language Bert

Our goal is to evaluate and to compare the performances of two different models, CANINE and a multilingual BERT model.

Settings for the experiment With using the HuggingFace Trainer, one can explicitly set the same settings for both CANINE[2] and mBERT[3] trainings. By using the same settings, one can ensure a fair comparison between the performance of the two models on the task. This helps to determine which model is more suitable for the needs while accounting for factors like training data, architecture, and input representation.

Same training set: We use the same dataset for training both models to make sure the differences in performance are due to differences in models, not data.

Tokenization changes: The only difference in the input is the tokenization. We use padding at 512 tokens per word for mBERT and padding at 2048 tokens per character for CANINE. This is because one word is approximately four characters long. In both cases, longer reviews are cut off.

Learning rate, optimizer, and weight decay: We use the same decreasing learning rate initially set at 2e-5, the same optimizer, and the same weight decay for both models.

Training duration: Both models are trained for 5100 steps, which is about 1.5 epochs.

Classification head: We use a dense layer to map the 768 pooled features to the 5 classes (1 class per star rating). A dropout layer with a probability of 0.1 is added right before the dense layer.

Review body only: The classification is based on the body of a review, not any additional information like the title.

Evaluation metrics: We use precision, recall, and F1 score as evaluation metrics to evaluate the performance of both models on the task.

By using these settings, we can compare the performance of CANINE and mBERT on the same task and determine which model is more suitable for our needs.

Results As seen in fig. 7, mBERT starts to perform much better from 600 steps onwards. It begins to overfit from step 3600, while CANINE hardly overfits from step 4200, which might be due to the limited amount of data. The training curve for CANINE is steeper compared to mBERT. Tab. 2 displays the evaluation metrics on the test set. mBERT outperforms CANINE in every metric and has a shorter training time.

[2]CANINE on HuggingFace
[3]mBERT on HuggingFace

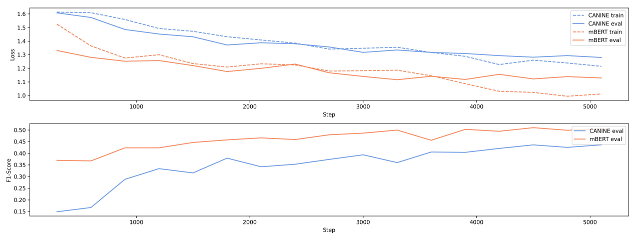

Figure 7. Cross-entropy (top) and F1-score on the validation set (bottom) seen during training both models

Model	Training Time	Cross Entropy	Precision	Recall	F1-score
CANINE	2h14	1.322	0.4161	0.4172	0.4153
mBert	**1h46**	**1.135**	**0.5006**	**0.4996**	**0.4980**

Table 2. Test metrics comparing Multi-lingual BERT and CANINE

Interpretation Contrary to our expectations, mBERT performs significantly better than CANINE. CANINE has 132 million parameters, while mBERT has 177 million parameters, which means that mBERT has greater learning capabilities. For a more accurate comparison, both models should have the same number of parameters. Another difference between the models is their structure, as explained in the CANINE section. The padding for CANINE can be set much higher, but increasing it to 2048 might reduce its learning capabilities. It is difficult to find CANINE and mBERT models with the same number of parameters, which would provide a more accurate comparison. Moreover, we should not forget that most a language model performances is obtained through pre-training on large (multilingual) dataset. Differences in this pre-training step may impact the performances obtained while

5. Conclusion

In conclusion, we tried to refine the CANINE model and compare it with a multilingual BERT model on a specific dataset. We trained and evaluated the models on the Tydi QA and Multilingual Amazon Reviews Corpus datasets, using appropriate metrics to measure their performance.

We fine-tuned a CANINE model on a question answering dataset to reproduce the results introduced by the authors. We were surprised to see that our model, trained a very small dataset, could achieve comparable results. On the Multilingual Amazon Reviews Corpus dataset, mBERT outperformed CANINE, achieving higher accuracy in predicting star ratings for product reviews. Training mBERT was also faster.

Unfortunately, our limited resources have restricted us in our results. It has been difficult, if not impossible, to train these types of models with large databases. The lack of access to Tensorboard also made it difficult to exploit the results. As a result, some of our training sessions had to be deliberately long in order to collect enough data for analysis and model improvement. For instance, to obtain the results of fig. 4 required 12 hours of computation on a single active session in Colab.

References

[1] Jonathan H. Clark, Eunsol Choi, Michael Collins, Dan Garrette, Tom Kwiatkowski, Vitaly Nikolaev, and Jennimaria Palomaki. TyDi QA: A benchmark for information-seeking question answering in typologically diverse languages. *Transactions of the Association for Computational Linguistics*, 2020.

[2] Jonathan H. Clark, Dan Garrette, Iulia Turc, and John Wieting. Canine: Pre-training an efficient tokenization-free encoder for language representation. *Transactions of the Association for Computational Linguistics*, 10:73–91, 2022.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018.

[4] Phillip Keung, Yichao Lu, György Szarvas, and Noah A. Smith. The multilingual amazon reviews corpus. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, 2020.