

# CANINE vs BERT on NLP Tasks

Dina EL ZEIN [dina.el-zein@ens-lyon.fr](mailto:dina.el-zein@ens-lyon.fr)

Alexandre MARTHE [alexandre.marthe@ens-lyon.fr](mailto:alexandre.marthe@ens-lyon.fr)

April 20, 2022

## 1 Introduction

This report is based on the work of J. Clark et al [1]. Pre-trained text encoders have superseded the state of the art on many natural language processing (henceforth NLP) tasks, yet they still require a tokenization step. In this paper, the idea is to build a free tokenizer encoder and vocabulary that works directly on character sequences, thus CANINE<sup>1</sup>. The code and checkpoints are available on GitHub at <https://github.com/google-research/language/tree/master/language/canine>.

Through this report, an overview of the model introduced in this article by J. Clark et al [1] is presented. Experiments on new datasets using CANINE are proposed with a direct comparison with mBERT [2]. The code to replicate the experiments is released at <https://github.com/dinalzein/CANINE-BERT-Experiments>

## 2 Method

CANINE, the character level model is simply implemented by taking as input a sequence of characters instead of subwords. The implementation consists of three main parts: (1) a downsampling function DOWN, (2) a primary encoder ENCODE, (3) an upsampling function UP. Given an input sequence of character embeddings  $e \in \mathbb{R}^{n \times d}$  with length  $n$  and dimensionality  $d$ :

$$Y_{seq} = UP(ENCODE(DOWN(e))) \quad (1)$$

such that  $Y_{seq} \in \mathbb{R}^{n \times d}$  is the final representation for sequence prediction tasks while for classification tasks, the model simply takes the zeroth element of the primary encoder:

$$y_{cls} = [ENCODE(DOWN(e))]_0 \quad (2)$$

The input representation to CANINE is a sequence of integers consisting of the codepoint integer of the characters. To represent all the unicode<sup>2</sup> characters using only a small number of parameters, they use different hash functions to embed these codepoints integers. This way, the model can learn representation for characters during fine-tuning never seen during pre-training.

Going through the implementation parts, firstly, for the downsampling function, an encoding of the characters using a single-layer block-wise local attention transformer has been done following by strided convolution to reduce the number of sequence positions to be similar to that of a word

---

<sup>1</sup>CANINE: Character Architecture with No tokenization In Neural Encoders.

<sup>2</sup>codepoint values are part of the Unicode Standard

piece model. Secondly, for the primary encoder, a deep transformer stack with different layers is applied to the downsampled output. Lastly, for the upsampling function, a reconstruction for the character-level output is done by concatenating the output of the original character transformer with the downsampled representation output by the deep transformer stack.

### 3 Data

**General Language Understanding Evaluation (GLUE)**<sup>3</sup> benchmark is a collection of natural language understanding tasks on different datasets [4]. We use the following datasets from this benchmark:

#### 3.1 Sentiment Analysis (SST-2)

SST-2<sup>4</sup> dataset [3] consists of 70042 sentences taken from movie reviews with human annotations of their sentiment. Each sample is linked with a label to tell whether or not the sentence is semantically correct. The dataset is split with 67349 sentences for training, 872 for validating, and 1821 for testing.

**Metric:** For evaluation, we use the accuracy metric which measures how many samples are correctly classified among all samples. The higher the value is, the better the result is.

**Task:** A binary classification task to predict whether the movie review is semantically correct or not.

**Setup:** To fine-tune on SST-2, we use a batch size of 16 for 5 epochs with learning rate =  $2e - 5$  and weight decay = 0.01. Then we select the best fine-tuning model on the training set and use it for evaluation.

#### 3.2 The Stanford Question Answering Dataset (SQuAD)

The Squad<sup>5</sup> dataset is an English-language Question Answering dataset of 98169 questions generated from 536 Wikipédia articles. Each sample consists in a question, a context, and an answer. There are 87599 training samples and 10570 validation ones.

**Metric:** For evaluation, we use the F1-score, which is the measure of overlap between the predicted answer and the correct answer, and is defined as the harmonic mean of the precision and recall.

**Task:** A question answering task where the answer is to be found among a specific context.

**Setup:** We pre-process the data to add the labels for the start and end position of the answer in the context. Afterwards, we fine-tune the model on the training dataset with the following parameters: a batch size of 4, 3 epochs, a learning rate of  $2e - 3e$ , and weight decay of 0.01. For the evaluation, we pre-process the data again to match logits and the characters in the context.

---

<sup>3</sup><https://gluebenchmark.com/leaderboard>

<sup>4</sup><https://nlp.stanford.edu/sentiment/index.html>

<sup>5</sup><https://rajpurkar.github.io/SQuAD-explorer/>

Model	Input	Training Loss	Evaluation Loss	Accuracy
BERT	Subwords	0.060	0.334	0.927
CANINE-S	Subwords	0.157	0.625	0.851
CANINE-C	Chars	0.169	0.572	0.856

Table 1: Comparison between BERT and CANINE on SST-2

Model	Input	Training Loss	Evaluation Loss	Exact Match	F1-score
BERT	Subwords	0.773	1.157	76.746	85.134
CANINE-S	Subwords	0.629	1.461	72.526	82.183
CANINE-C	Chars	0.664	1.355	72.375	82.300

Table 2: Comparison between BERT and CANINE on SQuAD

### 3.3 The Corpus of Linguistic Acceptability (CoLA)

CoLA<sup>6</sup> dataset [5] consists of 10657 English sentences taken from 23 different linguistics publications. Each sample in this dataset, which is a sequence of words, is associated with a label to tell whether this sentence is grammatically acceptable or not. From this dataset, 8551 sentences are used for training, 1043 sentences for validating, and 1063 for testing.

**Metric:** For evaluation, we use Matthews correlation coefficient metric which measures the association for two binary values and ranges from  $-1$  to  $1$ :  $1$  indicating a perfect match,  $-1$  indicating a perfect disagreement, and  $0$  indicates an uninformed guessing (similar to random) between the predicted label and the observed one.

**Task:** The task on this dataset is a binary classification task to predict whether an English sentence is grammatically correct or not.

**Setup:** To fine-tune on CoLA, we use the same setup as for SST-2.

## 4 Results

In this section we present CANINE and BERT fine-tuning results on the NLP tasks presented in Section 3.

To fine-tune using CANINE, we use checkpoints from the two models: **CANINE-C**<sup>7</sup> pre-trained with autoregressive character loss and **CANINE-S**<sup>8</sup> pre-trained with subword loss. Both models were pre-trained with 12-layers, 768-hidden, 12-heads, and 121M parameter. For BERT, we use the one trained on lower-cased English text that consists of 12-layers, 768-hidden, 12-heads, and 110M parameters.

Table 1 represents the results for a comparison between BERT and CANINE for SST-2 classification task. CANINE-C improves over CANINE-S by 0.005 negligible accuracy. BERT outperforms both CANINE-S and CANINE-C by  $+0.076$  and  $+0.071$  accuracy respectively.

<sup>6</sup><https://nyu-ml1.github.io/CoLA/>

<sup>7</sup><https://storage.googleapis.com/caninemodels/canine-c.zip>

<sup>8</sup><https://storage.googleapis.com/caninemodels/canine-s.zip>

Model	Input	Training Loss	Evaluation Loss	Matthews Correlation
BERT	Subwords	0.149	0.766	0.565
CANINE-S	Subwords	0.615	0.638	0
CANINE-C	Chars	0.567	0.668	0.064

Table 3: Comparison between BERT and CANINE on CoLA

Results presented in Table 2 represents the performance of fine-tuning CANINE and BERT on question answering task, SQuAD. BERT outperforms both CANINE-C and CANINE-S by +2.834 F1 and +2.951F1 respectively.

Table 3 represents the results for a comparison between BERT and CANINE for CoLA classification task. Fine-tuning using CANINE-S (CANINE with the subword loss) shows an inefficient performance as Matthews Correlation = 0 means that prediction is randomized. Using CANINE-C (CANINE with the character loss), the Matthews Correlation = 0.064, slightly better than CANINE-S but still not comparable to BERT. CANINE is not good for this task at all.

## 5 Conclusion

We presented a comparison between the state-of-the-art NLP models CANINE and BERT on 3 different NLP tasks: SQuAD, SST-2, and CoLA. In fact, BERT outperforms CANINE for all the NLP tasks presented in Section 3 in terms of F1 score, accuracy, and Matthews Correlation.

## References

- [1] Jonathan H Clark et al. “Canine: Pre-training an efficient tokenization-free encoder for language representation”. In: *arXiv preprint arXiv:2103.06874* (2021).
- [2] Jacob Devlin et al. “Bert: Pre-training of deep bidirectional transformers for language understanding”. In: *arXiv preprint arXiv:1810.04805* (2018).
- [3] Richard Socher et al. “Recursive deep models for semantic compositionality over a sentiment treebank”. In: *Proceedings of the 2013 conference on empirical methods in natural language processing*. 2013, pp. 1631–1642.
- [4] Alex Wang et al. “GLUE: A multi-task benchmark and analysis platform for natural language understanding”. In: *arXiv preprint arXiv:1804.07461* (2018).
- [5] Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. “Neural Network Acceptability Judgments”. In: *arXiv preprint arXiv:1805.12471* (2018).