Comparison between Encoder and Decoder-only Models for Sentiment Analysis

Domenico Plantamura, Eduardo David Lotto, Gabriele Fugagnoli, Manuel D'lterio Grazioli

1. Introduction

Sentiment analysis (SA) has long been a fundamental challenge in natural language processing. Historically, sentiment classification tasks relied on task-specific models. However, with the rise of decoder-only models, there is significant potential for their application to SA problems.

We want to answer the following question: given a setting with limited computing power, can decoder-only models give better performances in Sentiment Analysis than an encoder-based model? To do so we will compare the performances of a BERT model with the performances of Llama 3 on a Sentiment Analysis task. The objective is to perform this comparison, through a suitable dataset, and find if Llama represents an improvement on the widely used BERT. Furthermore, different prompt strategies will be explored. We especially want to address this topic from the point of view of a standard user who wants to perform this task and, having limited resources, wants to know which model is better and easier to use.

We selected the IMDB dataset, which includes 50,000 movie reviews labelled as either positive or negative. This dataset is a well-known and widely-used benchmark in the field of Natural Language Processing. By using a dataset that is both popular and readily accessible, we can ensure that our results will be comparable to those of other studies.

2. Transformers

Both BERT and LLama 3 are transformer-based models. In this section we will briefly describe the transformer architecture as outlined in the renowned paper "Attention Is All You Need."

The Transformer is a model architecture that avoids recurrence and instead relies entirely on the attention mechanism to draw global dependencies between input and output. The attention mechanism is the key part of the architecture.

2.1. Attention mechanism

An attention function can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key.

Instead of performing a single attention function, it is beneficial to project the queries, keys and values h times with different, learned linear projections, on which then performing the attention function in parallel. This technique is called multi-headed attention: it allows the model to jointly attend to information from different representation subspaces at different positions. Each head learns something different and it gives the model more representative power.

Essentially, Attention allows the selection of a small amount of important information in order to focus on these. The weight represents the importance of information. The goal is updating the embeddings using information from the context.

2.2. BERT

BERT (Bidirectional Encoder Representations from Transformers) is a pre-trained natural language processing model released by Google in 2018. At its core is the Transformer encoder, which enables unsupervised pre-training on a large corpus, followed by fine-tuning on various NLP tasks. BERT is a bidirectional deep learning model that simultaneously considers all words in context, allowing it to better understand the meaning of a sentence.

BERT's innovation lies in its use of a bi-directional Transformer encoder that can take into account context information in both left and right directions to capture richer language features.

BERT's pre-training tasks include Masked Language Modeling (MLM) and Next Sequence Prediction (NSP), which are used to learn vocabulary and sentence-level representations, respectively. MLM consists of randomly replacing some words in the input text with a special

symbol [MASK] and then asking the model to predict the obscured words. NSP is a binary classification task that gives two sentences and lets the model decide whether they are continuous or not.

The input sequence of BERT is composed of two sentences with a [SEP] token in between, and the initial "classification token" [CLS] that will later be used for prediction. Each token has a corresponding embedding, a segment embedding that identifies each sentence, and a position embedding to distinguish the position of each token (same as the *positional encoding* in the Transformer paper). All these embeddings are then summed up for each token.

In Sentence Pair Classification and Single Sentence Classification, the final state corresponding to [*CLS*] classification token is used as input for the additional layers that make the prediction.

2.3. **LLama 3**

Llama 3 is a large language model based on the attention mechanism. Llama models are trained on trillions of tokens and show that it is possible to train state-of-the-art models using publicly available datasets exclusively, without resorting to proprietary and inaccessible datasets. Llama is based on the transformer architecture with some key differences from the original architecture.

To improve training stability, pre-normalization is applied to the input of each transformer sub-layer instead of normalising the output. For what regards activation functions, the ReLU nonlinearity is replaced with the SwiGLU. Additionally, absolute positional embeddings are removed and replaced with rotary positional embeddings (RoPE) at each layer of the network.

The improvement between Llama 2 and Llama 3 comes from the use of a bigger training dataset, longer context length and the addition of group query attention to the smaller models.

3. **Encoder Based models vs decoder-only models in sentiment analysis**

As LLama is very recent, we searched the literature for papers comparing BERT and decoder-only models performances on Sentiment Analysis tasks. We found interesting results in the comparison between BERT and GPT-3.5.

3.1. **BERT vs GPT-3.5**

In the paper "Can ChatGPT Understand Too? A Comparative Study on ChatGPT and Fine-tuned BERT" it is shown that ChatGPT can achieve comparable or even better performances than existing LLMs on several generation tasks.
More specifically, ChatGPT shows robust performances in understanding, reasoning and generating relevant answers for sentiment classification; it achieves accuracy performances comparable to that of the BERT-base on sentiment analysis tasks. Furthermore, ChatGPT benefits significantly from advanced prompting strategies.

For what concerns BERT, fine-tuned models provide strong baseline performance on sentiment analysis with high accuracy. These models require large amounts of labelled data for fine-tuning to achieve optimal results. In terms of performance, BERT models perform well in structured classification tasks but may require additional fine-tuning for contextual understanding.

To conclude, ChatGPT, like BERT, shows strong capabilities in sentiment analysis, with a particular focus on inference and reasoning tasks.
Applying advanced prompting strategies to ChatGPT can further increase its performance, making it a valuable tool for sentiment analysis in various applications.

Performances on Sentiment Analysis task	
Method	Accuracy
BERT_base	88%
BERT_large	96%
roBERTa_large	96%
ChatGPT	92%

Comparison on Sentiment Analysis between ChatGPT and fine-tuned BERT-style models, for more details see reference [3]

4. **Experiments**

We implemented versions of BERT and Llama 3 to test their performances on the IMDb dataset. To get to a manageable computing time we subsampled the original dataset. We fine-tuned both models in different ways to obtain the best possible results given the limited resources of a standard user. Then we analysed them to make a comparison between these two different architectures.

4.1. **Llama 3**

We selected the 8 billion parameter chat version of Llama 3, tested the original pre-trained model's capabilities on the sentiment analysis task, and then fine-tuned it using the QLoRA method. After fine-tuning, we evaluated the performance improvements and tested various prompting strategies.

4.1.1. **QLoRA Fine Tuning**

Fine Tuning very large models is prohibitively expensive and recent quantization methods can reduce the memory footprint of LLMs. Parameter-Efficient Fine-Tuning (PEFT) techniques significantly reduce computational requirements by selectively fine-tuning small subsets of (additional) parameters, while keeping the majority of the pre-trained LLM parameters untouched. Among the PEFT techniques, there are LoRA (Low-Rank Adaptation) and QLoRA (Quantized Low-Rank Adaptation). We used the latter to fine-tune our model.

LoRA is a technique that fine-tunes a pre-trained language model by inserting low-rank matrices into each layer of the model and updating these additional parameters instead of all the weights of the model, which have a larger size. This technique offers several benefits: it reduces computational and memory overhead during fine-tuning, enables efficient adaptation of large models to specific tasks without updating all parameters, and conserves storage space by requiring storage for only a few parameters.

QLoRA, described in "QLoRA: Efficient Fine Tuning of Quantized LLMs", extends the idea of LoRA by incorporating quantization techniques, which implement the process of converting the model's parameters from a higher precision to a lower precision to reduce the model size. This leads to even lower memory and computational requirements and a faster inference due to reduced model size and lower precision arithmetic.

In particular, QLoRA provides two important implementations to reduce memory use without sacrificing performance: 4-bit NormalFloat and Double Quantization. In our experiments we used the 4-bit NormalFloat, an information-optimal quantization data type for normally distributed data that yields better empirical results than 4-bit Integers and 4-bit Floats, and avoided Double Quantization, which reduces the average memory footprint by quantizing also the quantization constants and saves an additional 0.37 bits per parameter.

The parameters used are the ones provided in "QLoRA: Efficient Fine Tuning of Quantized LLMs": learning rate (2e-4), max gradient norm (0.3), and warmup ratio (0.03). The latter specifies, along with the learning rate scheduler type ('cosine' in our experiments) the proportion of the total training steps during which the learning rate will gradually increase from zero to the initial learning rate value.

4.1.2. **Zero-Shot Prompting**

The first prompting strategy we employed is Zero-Shot Prompting: we did not provide Llama with examples or demonstrations to rely on for its classification output. Instead, we created a prompt that includes instructions for the model to understand what it needs to do and the movie

review it needs to classify. An example of a Zero-Shot Prompt given in input to our Llama model is:

“You are a sentiment classifier. For each given movie review, determine whether its sentiment is positive or negative. Return the sentiment label as either "positive" or "negative".

Now, analyse the following review:

Review: [*review text*]

Sentiment: “.

4.1.3. **Few-Shot Prompting**

On the fine tuned Llama 3 model we also implement two types of Few-Shot Prompting, so we provided some examples from which the model can perform an in-context learning, in order to get better performance. The examples serve as conditioning for the last review of the prompt where we want the model to generate a response. In particular we implemented a Two-Shot and a Four-Shot Prompting. The former includes one movie review per sentiment class, so one positive and one negative istance. The latter just doubles the number of examples, so we have two negative reviews and two positive reviews. An example of a Two-Shot Prompt given in input to our Llama model is:

“You are a sentiment classifier. For each given movie review, determine whether its sentiment is positive or negative. Return the sentiment label as either "positive" or "negative".

Here are some examples:

Review: [*positive review text*]

Sentiment: positive

Review: [*negative review text*]

Sentiment: negative

Now, analyse the following review:

Review: [*review text*]

Sentiment: “.

In order to build our prompts for Llama in the most correct and reliable manner we refered to the Meta guide “Prompting | How-to guides”.

4.1.4. **Performance analysis**

	Non fine-tuned model	**Fine-tuned model**	**Two-Shots prompt**	**Four-Shots prompt**
Accuracy	0.865	0.972	0.970	0.970
F1-score	0.91	0.97	0.97	0.97

It is possible to see that, in general, Llama obtains very good performances.

Also the non fine-tuned model of Llama gets pretty decent scores, with a remarkable accuracy of 0.865. The fine-tuned Llama enhances these performances, getting an accuracy of 0.972 and a F1-score of 0.97.

It is possible to see that using Few-Shot Prompting did not improve our fine-tuned model, in fact we obtained the very same F1-score and a slightly worse accuracy (0.970) both with the Two-Shot and the Four-Shot Prompting. We can say that the performances are practically identical. This probably happens because the IMDb dataset is relatively simple for modern models, and achieving high accuracy like 0.972 indicates that the model is already performing near the ceiling for this task. Small fluctuations around this high level of accuracy are expected and may not signify significant differences in model performance.

4.2. **BERT**

We first started with the “Bert for Sequence Classification" model available in the huggingface library. As the name suggests, this model is pre-trained on text classification tasks. After tokenizing the samples using the BERT Tokenizer, we fine-tuned the model using the Trainer class from huggingface. After the fine-tuning, we tested the new model on the test set and computed the accuracy scores and the confusion matrix.

The model was fine-tuned and tested on a subsample of the original dataset to match the computation time of the Llama model, getting a F1-score of 89%.

We then switched to the more recent and complex model Roberta. Roberta was created starting from the bert model with 12 layers, 12 attention heads and 125 million parameters. Due to difficulties encountered in the implementation phase we didn't use the bigger Roberta-large. To fine-tune this model we used Lora training. With this improved model we reached a f1 score of 94%.

Both models were quite balanced in their predictions; from the confusion matrix we were able to see that there wasn't any preference for a specific label.

4.2.1. **Performance analysis**

	Bert-base	**Roberta**
Accuracy	0.89	0.94
F1-score	0.89	0.94

From a review of the literature on the topic, we can say that the bigger Roberta-large model can achieve performances of about 96% accuracy in sentiment analysis tasks, getting very close to the performance of Llama 3.

5. **Conclusions**

We showed that both the tested architectures can achieve satisfying performances of around 96-97% accuracy on sentiment analysis tasks.

Surprisingly, the most interesting conclusions didn't come from the performance of the tested models but from the experience of implementing them.

From a practical point of view, the first important difference that became apparent between the two architectures was the number of parameters. Decoder only models such as Llama are constructed by billions of parameters, that means that even with lora optimized fine tuning, in cases where the computational resources are limited, these types of models can only be trained on small amounts of samples. This could be especially troublesome in cases where the sample domain is very diverse and/or large.

Another less obvious consideration to make is that decoder-only models are capable of executing numerous tasks without needing a change to their architecture. This means that once a user becomes acquainted with the procedures of fine-tuning these models, little new work needs to be done when switching to a new task. On the other end, encoder models need to have a specific architecture for each task (at least in the final layers) hence requiring the developer to search the best model for their problem among a plethora of pre-trained models available on platforms like huggingface, adding a overhead in the implementation and sometimes the need of expertise in a NLP task.

For these reasons, the choice between encoder-only and decoder-only architectures is highly dependent on the problem at hand and the limitations of the setting. The most important elements that influence the decision are:

- The diversity of the training set: if the problem requires training on a lot of samples, bigger architectures could be unfeasible to use.
- Memory and computational constraints: encoder models are significantly smaller and thus more efficient.
- Time sensitivity: encoder models, by being smaller, are faster.
- Range of tasks: decoder models are more flexible, making switching between different tasks easier.

In conclusion, we can say that there isn't an objectively better architecture and that, despite the recent fast growth in interest and capabilities of decoder-only models, we can envision a future where both architectures will be employed to solve NLP tasks in different scenarios.

References

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, et al. Attention is all you need [EB/OL]. https://arxiv.org/abs/1706.03762, 2017

[2] Yichao Wu, Zhengyu Jin, Chenxi Shi et al. Research on the Application of Deep Learning-based BERT Model in Sentiment Analysis.

https://doi.org/10.48550/arXiv.2403.08217, 2024

[3] Qihuang Zhong, Liang Ding, Juhua Liu et al.
Can ChatGPT Understand Too?
A Comparative Study on ChatGPT and Fine-tuned BERT.
https://doi.org/10.48550/arXiv.2302.10198, 2023

[4] Wenxuan Zhang, Yue Deng et al.
Sentiment Analysis in the Era of Large Language Models: A Reality Check.
https://doi.org/10.48550/arXiv.2305.15005, 2023

[5] Hugo Touvron, Thibaut Lavril et al.
LLaMA: Open and Efficient Foundation Language Models.
https://arxiv.org/abs/2302.13971, 2023

[6] Jacob Devlin, Ming-Wei Chang et al.
BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.
arXiv:1810.04805, 2018

[7] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman et al.
QLoRA: Efficient Finetuning of Quantized LLMs.
arXiv:2305.14314, 2023

[8] Meta Llama.
Prompting | How-to guides.
https://llama.meta.com/docs/how-to-guides/prompting/#prompting-using-zero--and-few-shot-learning

WORK PLAN

Objectives:

1) Report of at least 4 papers:
   - Research of the State-of-the-art solutions in this field.
   - Review of papers that compare BERT with Llama or, since the latter is very recent, other decoder-only models.
2) Find a suitable dataset on which we can run the models and perform our analysis.
3) Test BERT on the dataset.
4) Test Llama 3 and different prompting strategies.
5) Compare and investigate the most interesting results, finding patterns in the performances and pros and cons of each model.

Tasks 1) and 2) were performed by Domenico Plantamura and Manuel D'Alterio Grazioli.

Task 3) and 4) were performed by Gabriele Fugagnoli and Eduardo David Lotto.

Task 5) were performed by all the members of the group.

Domenico Plantamura:
- Research and comparison of papers suitable for our purpose: 10 h
- Finding a suitable dataset: 2 h
- Revision and resume of the selected papers : 30 h
- Analysis of the results and comparison: 1 h

Gabriele Fugagnoli:
- Review of literature and available models: 10h
- Implementation of models: 30-40h
- Analysis of the results and comparison: 1 h

Eduardo David Lotto:
- Review of literature and available models: 5 h
- Implementation of models: 30-40 h
- Analysis of the results and comparison: 1 h

Manuel D'Alterio Grazioli:
- Research and comparison of papers suitable for our purpose: 15h
- Finding a suitable dataset: 2h
- Revision and resume of the selected papers : 25h
- Analysis of the results and comparison: 1 h