Classifying the Emotional Polarity of Digital Communications Using Large Language Models

Vincent Qin

Lynbrook High School
vinqin2008@gmail.com

Abstract

Large Language Models (LLMs) employ deep learning algorithms to generalize patterns in data. Applying these LLMs to classification tasks can reduce the required labor and time. The research aims to fine-tune the LLM Llama 3.1 to correctly identify whether a chosen text message exhibits a positive or negative emotion. The goal of this procedure is to apply the fine-tuned LLM to large databases of text messages and locate users whose recent texts contain a large proportion of negative samples. This way, I can alert the users and direct them to help very early on. I chose the Stanford Sentiment Treebank v2 (SST-2) dataset. It mimics the emotional polarity of real texts with its even positive-negative sample distribution and its contextless format. I used the Unsloth framework and LoRa to significantly reduce the resources required during the fine-tuning process. I tested the model by taking SST-2's train split and inputting them individually into the trained model. Using this method, I found the Llama model to be highly accurate, with an accuracy of 94.8%. Interestingly, it had a high average Binary Cross-Entropy (BCE) Loss of 0.782 but achieved high accuracy. The testing against other models shows that the BCE Loss for sentiment analysis is not correlated to the actual accuracy of the model. From the results, I determined Llama 3.1 was the most suitable LLM for the sentiment analysis of large text databases.

Introduction[1]

Artificial intelligence (AI), which uses algorithms to have computers perform advanced tasks, has surged dramatically over the past three years. Specifically, Large Language Models (LLMs) have gained massive popularity due to their wide-ranging capabilities. They are especially strong in Natural Language Processing (NLP). As a result, many have created AI assistants. However, I applied the LLMs' NLP skills to a less common use: text analysis.

Machine Learning (ML) algorithms such as Ada Boost and Random Forests are meant to make predictions on data. The goal is to find a pattern that is hard to see for humans that can be extrapolated to a new testing dataset. However, such algorithms struggle with language-based data. This is due to the complexities of language that simple algorithms are unable to grasp. LLMs provide an elegant solution to this problem by harnessing complex deep-learning algorithms.

Language classification tasks have also become increasingly useful due to their low maintenance cost and reliability. They can create high-accuracy predictions on massive datasets at little to no cost.

My goal is to fine-tune the LLM Llama 3.1 (Dubey et al., 2024) to identify whether a given text exhibits a positive or negative emotion. I plan to apply my fine-tuned model to a database of digital communications like text messages and direct people with a lot of recent negative messages to resources for help. I selected the Stanford Sentiment Treebank v2 (SST-2) (Socher et al., 2013) dataset because the ratio of positive to negative messages on the internet was closely mirrored by its distribution. All the samples are also in a parse tree format, where the text is separated from its context. This is also reflective of text messages. I thus deemed SST-2 an appropriate fit.

Literature Review

In text classification, deep learning models learn to analyze and categorize text data by encoding linguistic patterns in layers of a neural network. A common approach involves using embeddings, where each word or subword in a text is transformed into a numerical vector that captures semantic meaning. These embeddings are then fed into a neural network to detect patterns across sequences of words. LLMs have become especially popular because of their ability to understand complex relationships in text. By processing tokens with attention mechanisms, LLMs can focus on relevant parts of the text to learn the nuanced dependencies, making them highly effective for tasks like sentiment analysis.

LLMs based on the Transformer (Vaswani et al., 2017) architecture use self-attention to weigh the importance of each word in a sentence relative to others. This allows the model to capture context and relationships across long sequences. Unlike Convolutional Neural Networks (CNNs), which focus on local patterns, or Recurrent Neural Networks (RNNs), which are prone to losing information over long sequences, transformers handle both short- and long-range dependencies effectively. This flexibility makes them powerful for interpreting emotion in text.

LLMs have had great success when fine-tuned to be chatbots for specific fields. For example, BioBERT (Lee et al., 2020) is a fine-tuned version of Google's BERT (Kenton et al., 2019) LLM that specializes in the biomedical field of science. Another successful model is MedGPT (Kraljevic et al., 2021). MedGPT is a fine-tuned version of OpenAI's GPT-2 (Radford et al., 2019) that can predict future ailments from a patient's hospital record. Since LLMs are trained on such large datasets, they contain a lot of information that is otherwise hard to find. The convenience of being able to ask BioBERT questions rather than search the web for related terms is beneficial for scientists of all levels. Predictive models are also especially useful because they can accurately find correlations between two things. In MedGPT's case, it helps by finding patients with a similar series of medical diagnoses, inferring that their conditions may be related.

Methodology

Dataset Description

The Stanford Sentiment Treebank is a supervised learning dataset for sentiment classification. The SST-2 dataset comes from the Stanford Sentiment Treebank (SST), built from movie reviews. SST-2 is the binary classification version of SST, where the sentiment of each sentence is labeled as either positive or negative. There are around 70,000 samples in total, and each sample is separated from its context, otherwise known as parse tree form. Each sample is labeled with 1 or 0, representing a positive or negative emotion.

The data was collected by crawling and aggregating movie reviews from Rotten Tomatoes. Each sentence was then manually annotated with sentiment labels. All sentences with neutral and fine-grained labels present in SST were removed. The reviews were cleaned and separated for sentiment analysis. I chose this dataset because it reflects the ratio of positive messages to negative messages online. This is useful for my future use case because the model will be fine-tuned on data that reflects its prediction.

Model Selection and Architecture

Llama 3.1 is a transformer-based model developed by Meta AI for NLP tasks, including binary classification, such as sentiment analysis on the SST-2 dataset. It follows the standard transformer architecture, using multi-head self-attention to capture relationships between words in a sentence. This structure is effective for sentiment classification, where understanding context is crucial for determining emotional polarity.

Llama comes in various sizes, with models ranging from 12 to 64 layers. Each layer includes attention heads and a feed-forward network. The Gaussian Error Linear Unit (GELU) activation function is used in each layer, facilitating information flow during training. For SST-2, the model takes tokenized movie review sentences as raw input and processes them through transformer layers, generating a final vector representation of the sentence.

The output layer uses a fully connected layer with sigmoid activation for binary classification to produce labels for positive or negative emotions. Llama 3 can be fine-tuned to predict on datasets like SST-2. It efficiently adapts to the dataset's requirements while maintaining fast inference speeds for handling varying sentence lengths.

Unsloth and Low-Rank Adaptation (LoRA) (Hu et al. 2021) are two techniques that can enhance the performance and efficiency of fine-tuning large language models like Llama. Unsloth is a library designed to optimize the training and inference pipeline and reduce unnecessary computation. When applied to Llama, it can streamline the training loop and data loading. This significantly reduces the time required to fine-tune the model. This is particularly beneficial for large models like Llama, where the sheer number of parameters can slow down training. Unsloth also speeds up the deployment process, making real-world applications more efficient.

LoRA, on the other hand, focuses on reducing the computational complexity of fine-tuning. Instead of updating all of Llama's parameters, LoRA introduces low-rank approximation by training only a subset of parameters while keeping the rest of the model frozen. This approach drastically reduces the memory and computation needed, making it possible to fine-tune Llama in resource-limited environments. Despite the reduced number of parameters, LoRA maintains model performance. This allows Llama to achieve high accuracy on tasks while converging faster during training.

Together, Unsloth and LoRA make Llama more efficient in training and deployment. Unsloth optimizes the overall pipeline, while LoRA reduces the cost of fine-tuning by minimizing parameter updates. This combination allows for faster training and more resource-efficient use of Llama, enabling high performance in tasks like sentiment analysis without requiring extensive resources.

Dataset Preprocessing and Augmentation

The preprocessing of SST-2 for fine-tuning involves tokenization and conversion into the Alpaca format. In the Alpaca format (Fig. 1), each data point consists of three fields: instruction, input, and output. For sentiment classification, the instruction is framed as "Classify the sentiment of the following text." The text refers to the sentence from the movie review, and the output is the label. Before this conversion, the dataset undergoes essential preprocessing tasks such as tokenization, which breaks the text into individual tokens, and parse tree formatting, where the text is standardized by converting it to lowercase, removing punctuation, and handling special characters. The transformation of the SST-2 dataset into a format suitable for Alpaca-based models allows for more structured training.

instruction string · *classes* 1 value	input string · *lengths* 2 268	output string · *classes* 2 values
Classify the sentiment of the following sentence.	hide new secretions from the parental units	negative
Classify the sentiment of the following sentence.	contains no wit , only labored gags	negative
Classify the sentiment of the following sentence.	that loves its characters and communicates something rather beautiful about human…	positive
Classify the sentiment of the following sentence.	remains utterly satisfied to remain the same throughout	negative
Classify the sentiment of the following sentence.	on the worst revenge-of-the-nerds clichés the filmmakers could dredge up	negative
Classify the sentiment of the following sentence.	that 's far too tragic to merit such superficial treatment	negative

Figure 1. An example of an Alpaca-formatted dataset.

Model Training Process

The training procedure for the Llama model using the Unsloth library follows a structured approach to optimize for binary text classification tasks like sentiment analysis on the SST-2 dataset. The model is trained with the AdamW optimizer, which is known for its efficiency in large-scale language models. A learning rate of 0.002 is used, which allows the model to learn at a pace that avoids overshooting while maintaining sufficient gradient updates. The batch size is set to 2 per device, and gradient accumulation steps are set to 4 to make the model perform updates after accumulating gradients over multiple steps to simulate a larger batch size.

The model is trained for 60 steps, with 5 warmup steps to adjust the learning rate and stabilize the training process gradually. The training arguments include 16-bit floating point precision (fp16) for faster computations. The logging steps are set to 1 to ensure frequent updates on model progress, and the evaluation steps are set to 10, providing frequent validation checks during training.

Weight decay is applied at a rate of 0.01 to prevent overfitting by penalizing large weights in the model. LoRA is used to reduce the number of trainable parameters, further enhancing the model's generalization and computational efficiency. LoRA dropout is set to 0, which means the model fully utilizes the adaptation without dropping any connections during training. The use of gradient checkpointing through Unsloth makes it more resource-efficient.

This setup maintains a balance between computational efficiency and performance, making it suitable for fine-tuning large models like Llama on tasks like sentiment analysis.

Experimental Results

Experimental Setup

I used Google Colab since it provides access to GPUs for running code in a cloud environment. I worked with the NVIDIA Tesla T4 GPU, allowing faster model training. To manage datasets, I used the Hugging Face Datasets library to store and load my dataset. This combination provided a robust way to handle data and run deep learning experiments without needing a local high-performance computing environment. The cloud computing environment ensures sufficient resources for the task (Fig. 2).

1.254 hours used for training.

4514.9623 seconds used for training.

75.25 minutes used for training.

Peak reserved memory = 11.504 GB.

Peak reserved memory for training = 5.52 GB.

Peak reserved memory % of max memory = 78.004 %.

Peak reserved memory for training % of max memory = 37.429 %.

Figure 2. Resource statistics.

The combination of the Colab environment, LoRA, and Unsloth is needed. To come to this conclusion, I ran the same tests on a laptop that had 16 GB of memory and 256 GB of disk space. The computing power of this laptop is middle of the spectrum and thus represents what the average user will have. When attempting to run the same code without LoRA and Unsloth, I was only able to download about 30% of Llama’s parameters before I ran out of disk space. After implementing Unsloth, I tried to run the code again but ran out of memory during the training process. This shows how both LoRA and Unsloth are essential to the training process. Colab is a nice add-on because it removes the requirement of expensive computational power for personal LLM training.

Performance Evaluation and Analysis

The evaluation of the model's performance was based on several metrics, including accuracy, precision, recall, and

the F1 score. A confusion matrix was used to analyze the model's predictions, providing insight into the number of true positives, true negatives, false positives, and false negatives. Precision measures the proportion of correctly predicted positive instances out of all predicted positives, while recall indicates how well the model identifies actual positives. The F1 score offers a harmonic mean of precision and recall, which is especially useful when dealing with imbalanced data.
To find the model's relative strength, I compared it to three other models renowned for their NLP skills: Gemma (Riviere et al., 2024), DistilBERT (Sahn et al., 2019), and Mistral (Jiang et al., 2023). By comparing it to other strong models, I can determine its relative strength and find the unique reasons that make it stronger than the rest.

Discussion of Findings

The model achieved a high accuracy of 94.8%, which is the highest out of all 4 models tested (DistilBERT: 94.6%, Mistral: 94.2%, Gemma: 93.8%). These statistics prove its robustness when handling different types of prompts. It also had a relatively short inference time of only 21 minutes compared to the average of the other three, which was 62.3 minutes. Additionally, it was very good at predicting deceivingly worded prompts. Fig. 3 includes two samples that are worded deceivingly. All models except Llama predicted incorrectly. However, Llama uniquely struggled with short prompts. It could not gain enough information from the short prompts after they were already removed from context. This proved that the amount of information contained within a sample was the most important factor in determining Llama's accuracy of a prediction.

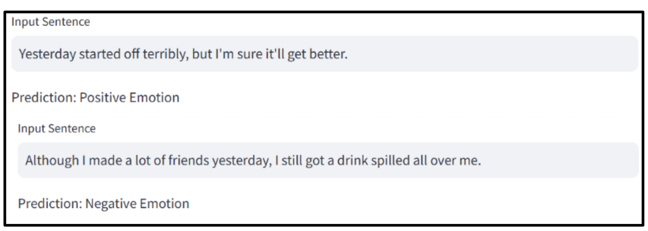

Figure 3. Predictions on two deceivingly worded samples.

The model also had a high BCE Loss, which usually indicates predictions that deviate from the sample. However, BCE Loss is an unsuitable metric to determine accuracy since the labels are binary, and therefore, any deviation from the correct answer is the maximum deviation. Thus, the high BCE Loss does not indicate that the model is inaccurate.
I also noticed that the sample length was an almost irrelevant factor for inference time. This is because the sample lengths of the SST-2 dataset were so short that they made little to no difference.

Conclusion

This study presented the application of fine-tuned large language models, specifically Llama 3.1, for binary text classification on the SST-2 dataset to identify positive or negative sentiments in text messages. The model achieved a high accuracy of 94.8%, which is better than other leading LLMs used for sentiment analysis. It demonstrated strong performance in analyzing both nuanced and straightforward emotional cues in the text. These findings confirm the effectiveness of transformer-based architectures in handling longer text data, even when context is absent. It also demonstrates the need for information within each sample for an effective output.
The high performance of the fine-tuned Llama model has practical implications in real-world applications where early detection of negative sentiment in text messages could be valuable, such as in mental health monitoring. The model's ability to process large datasets efficiently with tools like Unsloth and LoRA opens deployment possibilities in environments with limited computational resources. By analyzing sentiment trends in communications, this model can help identify and flag potential users experiencing prolonged negativity, enabling timely interventions or recommendations. Additionally, the methods and approaches used in this study can be adapted for other binary classification tasks involving short text data, such as spam detection or social media sentiment analysis.
One limitation of this study is the dataset itself. SST-2's binary sentiment labels may not capture the full range of human emotions expressed in text, potentially limiting the model's adaptability to real-world scenarios where more nuanced emotional gradations exist. Another constraint was using short, contextless samples, which affected the model's predictive accuracy on very short or ambiguous texts. Future research could explore using datasets with a wider range of emotions, including multi-label sentiment datasets, to improve the model's emotional sensitivity. Furthermore, experimenting with multi-task learning or hierarchical architectures could enhance the model's understanding of context and enable better handling of complex sentence structures. The limited resources of Google Colab heavily limited the training parameters. The lack of available GPU RAM and run time resulted in a suboptimal training environment. Without more resources, it is hard to determine the true potential of any LLM.
Refining these areas can contribute to a more robust application of Llama in sentiment and text classification.

References

Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; et al. 2024. The Llama 3 Herd of Models. Technical Report, arXiv:2407.21783.

Hu, E. J.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; and Chen, W. 2021. LoRA: Low-Rank Adaptation of Large Language Models. Technical Report, arXiv:2106.09685.

Jiang, A. Q.; Sablayrolles, A.; Mensch, A.; Bamford, C.; Chaplot, D. S.; de las Casas, D.; Bressand, F.; Lengyel, G.; Lample, G.; Saulnier, L.; et al. 2023. Mistral 7B. Technical Report, arXiv:2310.06825.

Kenton, J. D.; Chang, M.-W.; and Lee, K.; Toutanova, K. 2019. BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of NAACL-HLT, 1:2.

Kraljevic, Z.; Shek, A.; Bean, D.; Bendayan, R.; Teo, J.; and Dobson, R. 2021. MedGPT: Medical Concept Prediction from Clinical Narratives. Technical Report, arXiv:2107.03134.

Lee, J.; Yoon, W.; Kim, S.; Kim, D.; Kim, S.; So, C. H.; and Kang, J. 2020. BioBERT: A Pre-Trained Biomedical Language Representation Model for Biomedical Text Mining. Bioinformatics 36(4):1234-1240.

Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; and Sutskever, I. 2019. Language Models Are Unsupervised Multitask Learners. OpenAI Blog 1(8):9.

Riviere, M.; Pathak, S.; Sessa, P. G.; Hardin, C.; Bhupatiraju, S.; Hussenot, L.; Mesnard, T.; Shahriari, B.; Ramé, A.; Ferret, J.; et al. 2024. Gemma 2: Improving Open Language Models at a Practical Size. Technical Report, arXiv:2408.00118.

Socher, R.; Perelygin, A.; Wu, J.; Chuang, J.; Manning, C. D.; Ng, A. Y.; and Potts, C. 2013. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, 1631-1642.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention Is All You Need. Advances in Neural Information Processing Systems 30:5998-6008.