Lightweight Sentiment Analysis: Comparing Small LLMs, Large Models, and Traditional ML Approaches
Small NLP: CS 7643

Jaden Zwicker, Houshmand Abbaszadeh, Third Author
Georgia Institute of Technology
Atlanta, GA 30332
jzwicker3@gatech.edu, habbaszadeh6@gatech.edu, thirdauthoremail@gatech.edu

Abstract

Recent advancements in large language models (LLMs) have set unprecedented benchmarks in sentiment analysis, but their exceptional performance often comes at the cost of significant computational resources. This project investigates the viability of a lightweight, fine-tuned deep learning model capable of running on modest hardware, as a cost-effective alternative to models like BERT and GPT. Using IMDb's movie review dataset [2], we evaluate the performance of a 135M-parameter small LLM, "SmolLM," leveraging few-shot learning and fine-tuning to improve its sentiment analysis capabilities.

The project benchmarks SmolLM against state-of-the-art LLMs, non-transformer deep learning networks such as CNNs and LSTMs, and traditional machine learning approaches like Logistic Regression, Naive Bayes, and Random Forests. Performance metrics are derived from internal experimentation as well as from existing literature on this well-studied dataset.

Our goal is to demonstrate how fine-tuned small models can offer a competitive alternative to traditional methods and larger LLMs for relatively simple NLP tasks. In addition, we explore the diminishing returns of computationally intensive models, highlighting the potential to reduce deployment and operational costs without sacrificing performance. The outcomes of this work aim to guide practitioners in balancing accuracy with resource efficiency in sentiment analysis tasks.

1. Introduction/Background/Motivation

This project aims to compare the performance and computational costs of small and large language models for sentiment analysis on the IMDb movie reviews dataset [2]. The objective is to demonstrate that simpler models, when properly tuned, can deliver competitive performance without the need for expensive computational resources. This project will also explore how the small and large language models compare to traditional machine learning models like naive bayes, logistic regression, and random forest learners. By showcasing the effectiveness of lightweight models, we seek to address the widespread over-reliance on large, resource-intensive models and provide a sufficient and cost effective alternative.

(5 points) What did you try to do? What problem did you try to solve? Articulate your objectives using absolutely no jargon.

[1]

(5 points) How is it done today, and what are the limits of current practice?

(5 points) Who cares? If you are successful, what difference will it make?

1.1. Dataset

The dataset chosen to test the various models sentiment analysis capabilities was the IMDb's movie Review dataset [2]. This dataset contains strings of movie reviews, which was pulled from IMDb's website in 2011 making it a sample of the total amount of reviews they store. Each movie review sample is accompanied by the dataset's binary sentiment classification of Positive or Negative. The dataset contains 50000 total samples with 25000 being for training and 25000 for testing with an approximately 50/50 split between Positive and Negative classifications. These specifics of the dataset make it very balanced leading to it being well tested in literature and the perfect benchmark for basic sentiment analysis.

2. Approach

Mention the models were run on A100 with 40gb for consistency among tests HUGE NOTE, the order of pos vs neg in the asking of sentiment matters alot, first one listed tends to be the default models answer. Models

tend to say true more because they see it first in the prompt, but to counter this they do default to negative.

(10 points) What did you do exactly? How did you solve the problem? Why did you think it would be successful? Is anything new in your approach?

(5 points) What problems did you anticipate? What problems did you encounter? Did the very first thing you tried work?

Important: Mention any code repositories (with citations) or other sources that you used, and specifically what changes you made to them for your project.

3. Experiments and Results

SmolLM2-135M Zero-Shot adjusted accuracy when counting unknown predictins as false answers is 0.38

(10 points) How did you measure success? What experiments were used? What were the results, both quantitative and qualitative? Did you succeed? Did you fail? Why? Justify your reasons with arguments supported by evidence and data.

Important: This section should be rigorous and thorough. Present detailed information about decision you made, why you made them, and any evidence/experimentation to back them up. This is especially true if you leveraged existing architectures, pre-trained models, and code (i.e. do not just show results of fine-tuning a pre-trained model without any analysis, claims/evidence, and conclusions, as that tends to not make a strong project).

4. Other Sections

You are welcome to introduce additional sections or subsections, if required, to address the following questions in detail.

(5 points) Appropriate use of figures / tables / visualizations. Are the ideas presented with appropriate illustration? Are the results presented clearly; are the important differences illustrated?

(5 points) Overall clarity. Is the manuscript self-contained? Can a peer who has also taken Deep Learning understand all of the points addressed above? Is sufficient detail provided?

(5 points) Finally, points will be distributed based on your understanding of how your project relates to Deep Learning. Here are some questions to think about:

What was the structure of your problem? How did the structure of your model reflect the structure of your problem?

What parts of your model had learned parameters (e.g., convolution layers) and what parts did not (e.g., post-processing classifier probabilities into decisions)?

What representations of input and output did the neural network expect? How was the data pre/post-processed? What was the loss function?

Did the model overfit? How well did the approach generalize?

What hyperparameters did the model have? How were they chosen? How did they affect performance? What optimizer was used?

What Deep Learning framework did you use?

What existing code or models did you start with and what did those starting points provide?

Briefly discuss potential future work that the research community could focus on to make improvements in the direction of your project's topic.

5. Work Division

A summary of each authors contributions are provided in Table 3.

6. Appendix

add github repo with our code for tests

Model	Accuracy	Recall	Specificity	Precision	F-Score	% Positive	% Negative	# Unknown
SmolLM2-135M Zero-Shot	0.50	0.01	1.00	0.78	0.02	00.78%	99.22%	2993
SmolLM2-360M Zero-Shot	0.56	1.00	0.11	0.53	0.69	94.26%	05.74%	2
SmolLM2-1.7B Zero-Shot	0.72	0.99	0.45	0.65	0.78	77.01%	22.99%	14
SmolLM2-135M Few-Shot	0.59	0.80	0.37	0.56	0.66	71.86%	28.14%	0
SmolLM2-360M Few-Shot	0.66	0.99	0.33	0.60	0.75	82.74%	17.26%	0
SmolLM2-1.7B Few-Shot	0.81	0.99	0.62	0.73	0.84	68.48%	31.52%	0

Table 1. Default SmolLM2 Model's Results on Test Dataset

Model	Total Inference Time (s)	Average Inference Time (s)
SmolLM2-135M Zero-Shot	464.59	0.02
SmolLM2-360M Zero-Shot	499.90	0.02
SmolLM2-1.7B Zero-Shot	342.76	0.01
SmolLM2-135M Few-Shot	342.76	0.02
SmolLM2-360M Few-Shot	490.53	0.02
SmolLM2-1.7B Few-Shot	333.55	0.01

Table 2. Default SmolLM2 Model's Inference Times on Test Dataset

References

[1] HuggingFace. Smollm - blazingly fast and remarkably powerful. https://huggingface.co/blog/smollm, 2024. 1

[2] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. 1

Student Name	Contributed Aspects	Details
Jaden Zwicker	Default Model Experimentation	Performed various experiments on the 3 SmolLM2 models in their default configuration. Analyzed the results of these experiments to compare and contrast them with the larger LLMs to set a baseline of performance before fine tuning.
Team Member 2	Implementation and Analysis	Trained the LSTM of the encoder and analyzed the results. Analyzed effect of number of nodes in hidden state. Implemented Convolutional LSTM.
Team Member 3	Implementation and Analysis	Trained the LSTM of the encoder and analyzed the results. Analyzed effect of number of nodes in hidden state. Implemented Convolutional LSTM.

Table 3. Contributions of team members.