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

Jaden Zwicker, Houshmand Abbaszadeh, Brieuc Popper
Georgia Institute of Technology
jzwicker3@gatech.edu, habbaszadeh6@gatech.edu, bpopper3@gatech.edu

Abstract

Recent advancements in large language models (LLMs) have revolutionized sentiment analysis but often require significant computational resources. This project explores the viability of a lightweight, fine-tuned LLM, "SmolLM2" (135M parameters), as a cost-effective alternative to models like BERT and GPT. Using the IMDb movie review dataset [4], we evaluate SmolLM2's performance through few-shot learning and fine-tuning, benchmarking it against state-of-the-art LLMs and traditional machine learning models like Logistic Regression, Naive Bayes, and Random Forests. By comparing performance metrics from experiments and existing literature, we demonstrate how fine-tuned small LLMs can balance accuracy and resource efficiency, offering practical alternatives for sentiment analysis.

1. Introduction and 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 [4]. 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 over-reliance on large, resource-intensive models and provide a sufficient and cost effective alternative.

The IMDb dataset is a well-known dataset of movie reviews. Each movie review is approximately a few paragraphs, and the challenge is to perform binary sentiment analysis : predict if the movie review is *positive* or *negative*. A lot of researchers have tackled this problem and proposed very light models that achieve relatively good (more than 90%) accuracy on this benchmark. Using modern LLMs can achieve extremely impressive results (more than 94% accuracy), but at a great computational cost. We choose to explore the viability of a very small LLM (135M parameters, which is less than GPT-2 [5], and orders of magnitude less than SOTA LLMs) for sentiment analysis.

Furthermore, the authors of this small model *SmolLM2*, which we sought to fine tune, mentions that they were specifically designed for this type of use case [2]. In their analysis they mention that iPhone 15 Pros have 8GB of DRAM which is sufficient to run all three of their SmolLM2 Models [3]. With this they provide a memory footprint, which is in Figure 1, of the models in their default configurations proving just how little computational resources are needed to run these models which we hope to get to a point of comparable performance to larger industry grade LLMs.

All this being said, it is unlikely that a LLM is the best fitted solution for performing lightweight simple binary sentiment analysis. We focus on exploring specifically how well a small LLM can perform at this task, and how it matches up to other bigger LLMs, and then finally how it compares to other non-LLM techniques.

1.1. Why This Project Matters

Recently, some researchers have started to focus on extremely small-scale LLMs, that can run on-device even on some smartphones. These enable very low inference time with low computational requirements, and can be trained with less cost. It is unclear what use cases these LLMs can be used for however, as usually even 7 billion parameters LLMs are considered pretty bad at most tasks in 2024. This project aims to quantify how well SmolLM2, whose parameters and architecture is shown in Figure 2, can perform at simple sentiment analysis, how to set it up to achieve the best results, and to get an idea of how much of a cost reduction it enables.

Model	bf16	int8	int4
SmolLM-135M	269.03	162.87	109.78
SmolLM-360M	723.65	409.07	251.79
SmolLM-1.7B	3422.76	1812.14	1006.84

Figure 1. Memory footprint of SmolLM2models [2].

Model	#Param	#Layers	#Head	#KV-Head	Emb Dim	Hidden Dim	LR	BS
SmolLM-135M	135M	30	9	3	576	1536	3e-3	1M
SmolLM-360M	362M	32	15	5	960	2560	3e-3	1M
SmolLM-1.7B	1.71B	24	32	32	2048	8192	5e-4	2M

Figure 2. Architecture details of SmolLM2 models [2].

1.2. Dataset Selection

The dataset chosen to test the various models sentiment analysis capabilities was the IMDb's movie Review dataset [4]. 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 50,000 total samples with 25,000 being for training and 25,000 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. Given the datasets format as strings already and it being from the same creators as the models we were testing no pre-processing was needed outside of tokenizing each input prompt before feeding it into the language models.

2. Approach

2.1. SmolLM2 Choice

SmolLM2 [2], developed by Hugging Face, is a family of compact language models designed for efficiency and on-device performance. Available in three sizes—135M, 360M, and 1.7B parameters. Despite their smaller size, SmolLM2 models perform competitively on benchmarks for reasoning, coding, and general knowledge tasks, making them ideal for resource-constrained applications like mobile or edge computing. We chose SmolLM2 because it is a very recent and efficient model available in extremely small parameter sizes. The fact that it has multiple parameter size is also something we were looking for as it is possible to easily compare the impact of scale by switching between the different parameter sizes. We focused on the 135M model as the goal of this study is to see the viability of an extremely small LLM for simple NLP tasks.

2.2. Comparison Between LLMs, Traditional ML, and SmolLM2s

The goal of this experiment was to evaluate the performance and efficiency of various sentiment analysis techniques, highlighting alternatives to large language models (LLMs) that are more computationally affordable for resource-constrained environments. By comparing LLMs such as SmolLM2 and GPT-2 with traditional machine learning models like Naive Bayes, Random Forest, and Logistic Regression we aimed to identify resource-efficient solutions for sentiment analysis tasks. These traditional models were enhanced using TF-IDF to mitigate the influence of common terms. The experiment involved analyzing performance metrics such as accuracy and model size on the IMDb dataset, leveraging existing research to assess the feasibility of simpler, faster-to-train models for binary classification tasks like sentiment analysis.

A problem that we anticipated was the difficulty of accurately comparing models from different research papers due to variations in experimental methods, such as differing sample sizes, inconsistent evaluation metrics, and other methodological inconsistencies. However, this did not end up becoming an obstacle, since we were able to find sufficient documentation of sentiment analysis performed with various models all on the IMDb dataset from a single source with the same evaluation metrics, sample size, etc.

A problem that we encountered was that we had to read multiple papers before finding one that provided usable data about the performance metrics of each model, which could be used to compare the models to each other. Once we had found a paper with documented performance metrics, we were able to complete an analysis of the models and successfully identify alternatives to LLMs for sentiment analysis applications.

2.3. Generating SmolLM2 Benchmarks

To first investigate or compare the variety of models mentioned some initial tests and benchmarking had to be done. While traditional ML techniques for sentiment analysis and well documented LLMs have existing literature providing metrics for them running on the IMDb dataset [4] our choice for tuning a SmolLM2 model would require us to determine the baseline performance. Hence one of the first goals of this project was to test all three sizes of SmolLM2 models in both zero-shot and few-shot scenarios so later comparisons could better be made. This benchmarking was done on the 25,000 test samples from the dataset with each sample getting its own individual prompt to the model (no batching was used). The expected outcome of this experiment was that the greater parameter models would perform quite a bit better in terms of accuracy with few-shot learning also outperforming zero-shot in terms of simple accuracy.

An initial problem which was quickly realized from the

zero-shot testing was that the SmolLM2 models did not always return a clear Positive or Negative classification of the movie prompt. Since no previous examples of answering the question were given (few-shot did this) the smaller sized models such as SmolLM2-135M would simple repeat the question back or answer with random commonly used words such as 'I' or The repeatedly. To accommodate for this issue our initial approach had to be adjusted. Rather then parse the full reply that the models generated we took the top 50 most probable next words (logits) that the model generated and parsed through them to see if Positive of Negative appeared and if it did the first one was taken as the models answer. This methodology keeps the premise of sentiment analysis while taking into account that LLMs can give wordy responses making it hard to determine what sentiment they actually gave the prompt. Also, if the model was unable in those 50 most likely words to give an answer of either Positive of Negative this was treated as a default Negative answer however, we tracked these scenarios to bring up in later data points and analysis.

Another initial testing issue that arose, all of which allowed us to learn about the models better before fine tuning, was the dependence on the prompting method for determining the Models response. For example, if the prompt was phrased as follows: ***"Only Answer if this Movie Review is Positive or Negative:"*** then the smaller models would be more likely to select the first mentioned classification. So if the prompt asked Positive vs Negative then the model would be biased towards the Positive class. We could not determine a good way to elevate this for the zero-shot testing which is why we set the default class to Negative to assist in biases the choice the other direction. For the more advanced larger models or for the few-shot prompts this was less of an issue but during testing some bias could have been shown to the Positive class.

Finally, for the few-shot prompting of the models 3 example reviews were used before appending the given sample to the end of them. The 3 example reviews were generated by us and were purposely kept short for the sake of the models interpretation of them, they can be seen in Table 1.

2.4. Fine-tuning SmolLM2-135M

In addition to testing out SmolLM2 in zero-shot and few-shot settings, we decided to fine-tune it specifically for the IMDb benchmark. Our approach adapts the pretrained language model for binary sentiment classification (positive/negative), by replacing the original language model's head with a custom classification head. The SmolLM2 weights were frozen, and only the final custom classification head was trained. This aims to leverage SmolLM2's knowledge about language in general, as the embeddings in the last layer probably have rich context-aware information gained from the successive attention blocks in the Transformer architecture.

We needed to decide which token's embedding to use as the input for the final layer (the layer with trainable weights). A common approach when fine-tuning LLMs is to use the embedding of the very first token in the sequence, as this special token is designed to encapsulate information about the entire sequence. In our case, the sequence was the tokenized paragraph of the movie review. However, in our experiments, using the embedding of the first token did not yield good results. In the end we chose a prompt resembling a few-shot prompt, structured as shown below:

"example movie review" ; this movie review is : *positive* ; *"example moview review 2"* ; this movie review is : *negative* ; *"movie review to do inference on"* ; this movie **review** is :

We then used the embedding from the token "review" (highlighted in bold) from the last layer of the language model and used this as the input for the custom classification head.

3. Experiments and Results

3.1. Testing Default SmolLM2 Models

To compare the various models, we first benchmarked the 3 types of SmolLM2 models. All three; 135M, 360M, and 1.7B parameter SmolLM2 models were tested on the test section of the IMDb dataset [4] with both zero-shot methods and few-shot methods. The results of these experiments are given in Table 2 with the models overall accuracy being shown alongside the distribution of classifications which the models made over the prompts. We also report, for each scenario, the percentage of predictions classified as positive and negative by the model, noting that with a balanced dataset, this should ideally be 50/50. More detailed metrics such as Recall, Specificity, Precision, and F-Score were calculated but are not as necessary to the discussion of comparing these models to the other methods of sentiment analysis due to the dataset being binary and balanced.

To first analyze the results of the experiment it must be mentioned again that the models had a default sentiment classification of Negative. This is to explain why the percentages of Positive and Negative add up to 100% yet the models had some choices which were Unknown (the model did not classify). All Unknown predictions were defaulted to Negative for the sake of consistency among the metrics and this did not make a noticeable difference in all of the models besides SmolLM2-135M Zero-Shot. The smallest of the models when using the least advanced method of prompting had many glaring issues, which we sought to fix, but the first of which was its inability to generate a valid classification for **2,993** samples. When taking into account its lack of classification on the overall accuracy of the model

Few-shot Prompt 1	Movie Review: I loved this movie ! So good plot ! Only Answer if this Movie Review is Positive or Negative: Positive
Few-shot Prompt 2	Movie Review: I hated this, could be a lot better Only Answer if this Movie Review is Positive or Negative: Negative
Few-shot Prompt 3	Movie Review: This move was so good I would recommend to all my friends! Only Answer if this Movie Review is Positive or Negative: Positive

Table 1. Few-shot prompts used to test default SmolLM2 models.

Model	Accuracy	% Positive	% Negative	# Unknown
SmolLM2-135M Zero-Shot	0.50	00.78%	99.22%	2993
SmolLM2-360M Zero-Shot	0.56	94.26%	05.74%	2
SmolLM2-1.7B Zero-Shot	0.72	77.01%	22.99%	14
SmolLM2-135M Few-Shot	0.59	71.86%	28.14%	0
SmolLM2-360M Few-Shot	0.66	82.74%	17.26%	0
SmolLM2-1.7B Few-Shot	0.81	68.48%	31.52%	0

Table 2. Default SmolLM2 Model's Performance Metrics on IMDb Test Dataset

its accuracy goes from **0.5** to **0.38** making it by far the worst performing model in this project.

Furthermore, the SmolLM2-135M Zero-Shot model was the only one to pick majority Negative for the classification at 99.22% meaning its accuracy of 0.5 or 0.38 was not achieved by any smart methods which would generalize but more so by just picking one of the binary classification options. The reasoning for its preference of the Negative classification over the Positive was an outlier as-well since all other models preferred the positive classification greatly with that preference diminishing as the models went up in parameter size and ability to digest the problem at hand. During testing of the models, as mentioned in the approach, they had a tendency to prefer the classification which was first mentioned to them in the prompt. This effected the zero-shot models more however the SmolLM2-135M model really did just break out of the curve in terms of its predictions or lack their of.

Overall the experiment was successful as demonstrating a general trend among the models and providing some baseline results for the selected language models. For all of the other models they show the expected trend of few-shot prompting adding about a 10% increase in performance with the increase in parameters being the largest driving factor in accuracy increases with the best performing model being the SmolLM2-1.7B Few-Shot at 0.81 accuracy. This model also had the most even split between Positive and Negative classifications showing its ability to better understand the prompts being posed. While it still had a bias towards the Positive classification (perhaps because the few-shot prompt had two positive examples for one negative example) it was able to nearly perfectly predict any Positive prompt correctly while suffering in terms of accuracy due to false positives rather than false negatives. From this benchmarking we took away the importance of few-shot learning for our fine-tuned model and learned many lessons in how the prompts to the language models had to be structured. This also gave a good goal of potential performance to chase as we sought out to beat the best performing default model by using the smallest of the 3 with a Fine-tuned SmolLM23-135M.

3.2. Comparison Between LLMs, Traditional ML, and SmolLM2s

To start the comparison between the models, the final results of this experiment are shown in table 3. The Naive Bayes model was determined to be a satisfactory alternative to LLMs for simple sentiment analysis tasks, and for users who are okay with lower accuracy in exchange for significantly reduced memory usage and training time. Naive Bayes models store conditional probabilities for each feature/class, resulting in their memory size increasing linearly with the number of features/classes. Compared to LLMs, whose size depends on their millions or billions of parameters, Naive Bayes models are significantly smaller. In addition, Naive Bayes models are simple to train, training in just O(n*d), where n is the number of samples, and d is the number of features. This is due to the relatively simple calculations needed to compute the conditional probabilities of words and classes for each sentiment sample. Its performance on complex textual samples is hindered by its interpretation of each word as independent, resulting in erroneous classifications on samples that have conflicting word/sentiment probabilities. For example, a review that says "This movie was so good it made me want to ignore the whole thing and scroll on my phone instead" might be incorrectly classified as a positive review due to the high weight that "good" places on the positive review class probability. However, for simple texts such as those in the IMDb dataset, Naive Bayes achieved decent results: 0.8228 accuracy, 0.8285 precision, and 0.8174 recall [6]. These results indicate that the Naive Bayes model can correctly assign the correct sentiment class most of the time, but still has significant error.

The Random Forest model was determined to be a dissatisfactory alternative to LLMs due to its relatively high memory usage and training times, and no significant improvement in results. Random Forests can be large due

Model with TF-IDF	Accuracy	Precision	Recall	F-Score	AUC
Naive Bayes	0.8228	0.8285	0.8174	0.823	0.85
Random Forest	0.8562	0.8597	0.8539	0.8568	0.93
Logistic Regression	0.8914	0.882	0.9055	0.8936	0.96

Table 3. ML techniques Performance on IMDb Dataset

to their size multiplicatively increasing with the number of trees, nodes, and features. In addition, Random Forests are trained in a time of O(t*s*f*d), where t is the number of trees, s is the number of samples, f is the number of features, and d is the average tree depth. Training time can be significantly increased with the size of the forest/dataset, and is lengthy when compared to Naive Bayes models, or other, similarly less complex models. When run on the IMDb dataset, the Random Forest model did not achieve significantly higher results than the Naive Bayes: 0.8562 accuracy, 0.8597 precision, 0.8539 recall [6]. While smaller and faster to train than LLMs, Random Forests are not satisfactory alternatives due to still demanding relatively high memory and time resources, without the significant increase in results.

Logistic Regression was determined to be a good alternative to LLMs due to its high results on the IMDb dataset, and its satisfactory size and training times when compared to Random Forests. Logistic Regression's memory needs are dependent solely on the number of features, where Random Forests must account for many trees, which may have many branching subtrees/nodes. In addition, the Logistic Regression model trains in a time of O(d*k), where d is the number of features, and k is the number of iterations. While slower to train than Naive Bayes, the significantly higher results make it a comparable choice. When run on the IMDb dataset, the Logistic Regression model achieved high results: 0.8914 accuracy, 0.882 precision, and 0.9055 recall [6]. These results are comparable to LLMs, especially when the significantly lower memory requirements and training times of the Logistic Regression model is considered. For users who demand high results, and have decent resources for memory and training time, the Logistic Regression model is a good alternative to LLMs.

SmolLM2-135M was determined to be a good alternative to LLMs for sentiment analysis tasks due to its high accuracy. However, it is a less ideal alternative compared to Logistic Regression due to its slower training time and larger model size. SmolLM2-135M achieved 0.892 accuracy on the IMDb dataset performing sentiment analysis. This is comparable to both Logistic Regression and LLMs. SmolLM2-135M is faster to train than LLMs due to having only 135 million parameters compared to ChatGPT's 1.7 billion. In addition, having significantly less parameters means that SmolLM2-135M is much smaller in size and requires less memory resources for use on sentiment analysis tasks. Being able to achieve high accuracy results within 2 percentage points of LLMs, and being much smaller and faster to train, make SmolLM2-135M a good alternative to LLMs for sentiment analysis applications. However, SmolLM2-135M is a neural network based model, and results in a computationally intensive training process when compared to Logistic Regression. This, in addition to Logistic Regression's comparatively small size, make it a more suitable choice for a LLM alternative in most sentiment analysis use cases.

This experiment was successful in identifying alternative models to LLMs for sentiment analysis applications. Logistics regression with TF-IDF was determined to be the ideal alternative, due to its high accuracy results, relatively fast training times, and compact model size. Logistic Regression performed very well in terms of accuracy, reaching within 2 percentage points of LLMs. In most scenarios, Logistic Regression will provide comparable performance at a much lower cost. Naive Bayes was the fastest model to train and smallest in size, but fell 10 percentage points from LLMs in terms of classification accuracy. In use cases where speed and space are a tighter constraint than accuracy, Naive Bayes is a good alternative to LLMs. SmolLM2-135M delivered accuracy results equivalent to Logistic Regression but was much slower to train and significantly larger in size, making it a less favorable alternative to LLMs when compared to Logistic Regression. Random Forests proved to be a poor alternative due to their high training times and larger size relative to other alternatives, without achieving competitive accuracy. While these lightweight models are effective for general sentiment analysis tasks, there are scenarios where LLMs offer distinct advantages. For instance, LLMs excel in understanding nuanced language features such as sarcasm, irony, or context-dependent sentiment shifts, which simpler models may struggle to capture. In conclusion, we can confidently say that lightweight, high-performing models are viable alternatives for sentiment analysis applications, offering efficient solutions without the resource demands of LLMs. Among these, Logistic Regression with TF-IDF stands out as the most efficient option, balancing accuracy, speed, and compactness.

3.3. Results of Fine-tuning SmolLM2-135M

We used the experimental setup described in section 2.4. We trained for a total of 10 epochs using the whole train-

ing dataset, with a learning rate of 1e-4, on 4 GPUs with a batch size of 90/GPU. The network was made of a linear layer from the 576-dimensional embedding of a token from SmolLM2's last layer, which was fed through a linear layer down to 384 neurons, which go through a GELU activation, then a batch normalization, then to the final layer made of 2 neurons (one for positive, one for negative). Cross entropy loss was used for this binary classification task. Dropout (0.2) and weight decay (0.01) were added for regularization.

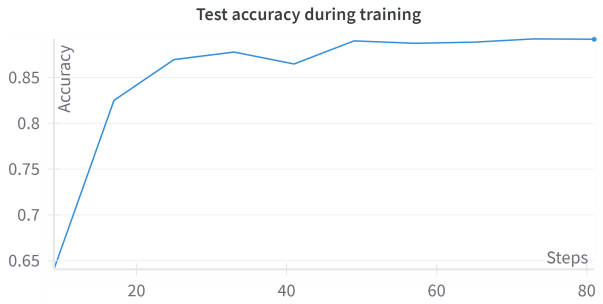

Figure 3. Fine-tuning SmolLM2 for 10 epochs with final accuracy reaching 89.2%

The final run we kept reaches 89.2% accuracy on IMDb's test set, as shown on figure 3. We tried out different architectures (changing the classification head, and also changing what was given as an input the this head). All the tests we tried with the input as described in 2.4 reached more than 88% accuracy. However the initial tests we ran ranged from 60% to 80% accuracy, when using the raw sequence and the very first or very last token of this raw sequence as the input for the classification head (this was before coming up with the prompt structure discussed in 2.4).

We believe it is achievable with careful hyperparameter tuning and architectural design, and perhaps some ensembling, to reach 90% with SmolLM2-135B. It is interesting to note that the authors of GPT-2 [5] reached an accuracy of 92.36% when fine-tuning their model. GPT-2 has a bigger parameters count than SmolLM2 however (1.5B), which could explain why it's fine-tuned accuracy is a bit better.

4. Conclusion

In this project, we evaluated several smaller language models and ultimately selected the SmolLM2 model for fine-tuning. SmolLM2 stands out as a recent and highly efficient option, available in a range of parameter sizes, the sizes that we focused on were the 135M, 360M, and 1.7B models.

First we tested and evaluated these models using both zero-shot and few-shot approaches and we successfully demonstrated an expected general trend that few shot prompting adds about a 10% increase in performance. Our analysis also revealed that increasing parameter size was a very significant factor in improving accuracy.

We also researched various different traditional machine learning methods for sentiment analysis, such as Naive Bayes, Random Forest classifiers, and Logistic Regression models. Our analysis of the different machine learning methods demonstrated that traditional machine learning methods have a strong performance without the expensive resources of LLMs. The most competitive traditional machine learning model was the Logistic Regression with TF-IDF with an accuracy of 89.14%.

We also experimented with the SmolLM2-135M's to see if we were able to fine tune the model well enough to be competitive to larger models. Our fine tuned SmolLM2-135M was able to achieve an accuracy of 89.2%, which is fairly competitive to GPT 2's accuracy of 92.36%. This experiment highlights the importance of fine tuning, particularly for smaller models, and it also highlights their potential to deliver high quality results with significantly less resources.

In conclusion, traditional machine learning models, specifically the Logistic Regression model with TF-IDF gives us the most competitive performance with fairly low costs and accuracy results that are comparable to the SmolLM2. This means that for very cheap but efficient sentiment analysis, LLMs are not the best choice. This is not that surprising as LLMs are trained to predict the most likely next token from a sequence, and so even though they have some language understanding and can do well at sentiment analysis, they are fundamentally not going to be able to compete with models designed from the start to just be cheap in compute and do sentiment analysis. In a sense, a lot of compute is "wasted" by using a LLM for a simple binary sentiment analysis task, compared to simpler models that still reach impressive accuracies.

That being said, LLMs are still be extremely valuable for sentiment analysis! This is because despite being computationally expensive, larger models shine as they can provide extremely high accuracies [1] on these simple sentiment analysis tasks. These bigger models have a fine-grained understanding of very subtle language nuances, and can pick up on things such as irony which can't be picked up by simple TF-IDF based models which are mostly based on word count. They can also be very useful for more difficult sentiment analysis problems, for instance when the sentiment analysis is no longer binary classification, but has 10 classes, or a continuous spectrum of classes.

5. Work Division

A summary of each authors contributions are provided in Table 4.

References

[1] Papers with code. https://paperswithcode.com/sota/sentiment-analysis-on-imdb. Accessed: 2024-12-11. 6

[2] HuggingFace. Smollm - blazingly fast and remarkably powerful. https://huggingface.co/blog/smollm, 2024. 1, 2

[3] HuggingFace. Smollm2. https://huggingface.co/collections/HuggingFaceTB/smollm2-6723884218bcda64b34d7db9, 2024. State-of-the-art compact LLMs for on-device applications: 1.7B, 360M, 135M. 1

[4] 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, 2, 3

[5] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019. 1, 6

[6] Vidushi Bansal Shweta Upadhyay Sandesh Tripathi, Ritu Mehrotra. Analyzing sentiment using imdb dataset, 2020. Page linking to ML techniques. 4, 5

6. A. Project Code Repository

The Github repository containing the experiments mentioned throughout the report alongside the default and tuned models can be found here.

Student Name	Contributed Aspects	Details
Jaden Zwicker	Default Model Experimentation & Analysis	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 and to set a baseline of performance before fine tuning.
Houshmand Abbaszadeh	Analysis & Comparison of Various ML Techniques	Performed an experiment to compare different ML techniques for sentiment analysis applications, successfully identifying alternative models to LLMs for sentiment analysis. Analyzed and documented the results of the experiment, as well as contributing to the abstract, introduction, and approach sections of the paper.
Brieuc Popper	SmolLM2 finetuning, general coding tasks	Designed and ran the fine-tuning experiments with SmolLM2-135M. Helped team setup on PACE ICE GPUs and all around Python Code with *transformers* library to deal with LLMs

Table 4. Contributions of team members.