Fine-tuning Large Language Models with Sharpness-Aware Minimization

Veniamin Veselovskyy, Cezary Januszek, Anne Gritto
Project in Optimization for Machine Learning, Department of Computer Science, EPFL, Switzerland

***Abstract*—The interest in developing and improving Large Language Models (LLMs) has increased considerably, while new methods such as synthetic data generation came to reinforce the traditional training of these models. With these heavily over-parameterized models, a need for better generalization is present. A recently proposed optimization algorithm, Sharpness-Aware Minimization (SAM), focusing on loss function convergence to flatter minima, empirically proved to show improved model performance in the computer vision domain. Motivated by these findings, we want to explore whether SAM brings similar improvement when fine-tuning LLMs with synthetic data.**

I. Introduction

Synthetic data generation has been shown to be a promising direction to better study constructs in text [1]. Such synthetic text data can then be used to fine-tune Large Language Models (LLMs) to study textual constructs on mass [2]. This, however, often leads to the models overfitting on the artifacts that are present in the generated text [3].

We hypothesize that there are two reasons why synthetically generated text may not be effective for training smaller-capacity classifiers. First, the data may simply be off due to specific biases present in the decodings——for example, when asking ChatGPT to write a sarcastic text it starts most with "Oh wow,...". Previous work has explored methods for enforcing diversity in RLHF models [3]. In this paper, we focus instead on a second possibility for *why* these models may not generalize effectively from the synthetic data: the optimization model locates sharp minima instead of shallow minima (i.e., the issue comes directly from the optimization model). Consequently, we explore if Sharpness-Aware Minimization (SAM) can help improve the generalization of fine-tuned LLMs with synthetic data, as suggested in past work [4].

To test this, we use a human-annotated dataset of sarcastic texts, and generate a new set of synthetic texts from it, illustrated in Figure 1. We then finetune a lower capacity encoder-based model on both the synthetic and real texts with and without SAM. If the increase derived from SAM on the synthetic text is higher than the increase derived from SAM on the real text, then we argue this means that the model helps generalize on synthetic data. In particular, our main research question is **RQ 1:** does fine-tuning a LLM with a SAM optimizer lead to better generalizability of synthetic text. We find that SAM does not lead to improved generalizability from synthetic data to real data.

II. Related Work

Sharpness-Aware Minimization was extensively presented by *Foret et al. (2021)* [5], as an optimization algorithm for simultaneous minimization of loss function value and sharpness. The authors of the paper argued that in modern over-parameterized models, typical optimization approach can result in sub-optimal results on test data. For an improved model generalization, they proposed to look for flatter minima in the loss function landscape. In fact, a positive correlation between the training loss sharpness and generalization error was observed in previous work [6], and a later empirical study showed that sharpness-based measures have actually the highest correlation with generalization [7]. *Bahri et al. (2022)* [4] extended *Foret et al.*'s work by suggesting that SAM can improve language models generalization without much computational overhead. They concluded that SAM used for fine-tuning of text-to-text transformers improves significantly test performance at the cost of around 25% extra compute. Secondly, the generalization improvement is even higher for data-limited tasks. This last aspect is further investigated by *Adriushchenko & Flammarion (2022)* [8], who also challenge the understanding of SAM success in previous works, which they label as incomplete. Additionally, they study the implicit bias of SAM and prove its convergence for non-convex objectives in a stochastic setting.

In this paper, we follow and extend these past works by exploring if Sharpness-Aware Minimization can improve generalization of LLMs when fine-tuning with real and synthetic data.

Review of Sharpness-Aware Minimization. First of all, we review the SAM algorithm. This optimization procedure seeks parameters w lying in neighbourhoods having uniformly low training loss - equivalently, it minimizes at the same time, both loss value and loss sharpness. To achieve this, SAM modifies the traditional stochastic gradient descent step, such that at every iteration a worst-case adversarial (ascent) point is computed and the gradient is taken at this point rather than at the current iterate. This results in formulating a min-max optimization problem on which SGD can be performed.

First, the notion of *m-sharpness* [8] is defined as the average of the sharpness computed over all batches S of size m from the training set S_{train}. This relates SAM directly to mini-batch SGD with batches of the same size m.

$$s(w, S) = \max_{\|\epsilon\|_2 \leq \rho} \frac{1}{m} \sum_{i \in S} l_i(w + \epsilon) - l_i(w)$$

Then, the corresponding m-SAM objective is defined as:

$$\min_{w\in\mathbb{R}^{|w|}} \sum_{|S|=m} \max_{\|\epsilon\|_2\leq\rho} \sum_{i\in S} l_i(w+\epsilon)$$

which is equivalent to

$$\min_{w\in\mathbb{R}^{|w|}} \sum_{|S|=m} s(w,S) + l_i(w)$$

Therefore the objective above clearly minimizes both the loss value and loss sharpness. Finding the optimal $\epsilon*$ is challenging and can be obtained using a first-order approximation resulting in

$$\hat{\epsilon}(w) = \rho \frac{\nabla_w L(w)}{\|\nabla_w L(w)\|_2}$$

which is just a scaling of the loss gradient at the current weights [4]. The ascent point is computed as

$$w_{adv} = w + \hat{\epsilon}(w)$$

That is the **SAM first step**. The **second step of SAM** then updates the original weights with the loss gradient at w_{adv}:

$$w_{t+1} = w_t - \gamma_t \nabla_w L(w)|w_{adv}$$

It is worth noting that SAM has a single hyperparameter ρ, which is the size of the neighbourhood and step taken along the unit adversarial gradient vector. Moreover, the use of worst-case perturbation and low m value can be essential for generalization improvement. The performance impact of random perturbations is marginal and lower m leads to more accurate maximization in the SAM first-step and better regularizing effect of batch normalization [8]. To make SAM's parameter ρ invariant to the scale of the model parameters an adaptive version of it was proposed - the Adaptive Sharpness-Aware Minimization (ASAM) [9].

In our implementation, we use this adaptive version of m-SAM presented before, to which we refer as SAM for simplicity, throughout the rest of this paper.

Synthetic data generation. Data augmentation with synthetic datasets can enhance model performance across various Natural Language Processing (NLP) tasks in settings characterized by low resources or imbalance. These tasks span relation extraction [10], sarcasm detection [11], translation [12], and sentiment analysis [13].

The recent employment of synthetic datasets for data creation is noteworthy. For instance, Eldan et al. [14] employed LLMs to construct "Tiny Stories," demonstrating the potential of compact language models to learn and mimic the language of children aged between 2 to 3 years. Furthermore, Josifoski et al. [15] explored the use of GPT-3 for generating text based on sampled knowledge graph triplets. They subsequently fine-tuned a model entirely on this synthetic data, an experiment that underscored a noticeable dissimilarity between the artificially created data and authentic human data.

III. Data & Methodology

Data. We use the *sarcasm detection* dataset from the SemEval-2022 Task 6 [16]. The train set includes over two thousand self-disclosed instances of sarcasm being shared on Twitter. The reason we choose sarcasm is because it is an inherently difficult task to annotate, and construct to capture. Sarcastic texts are highly context-specific and ambiguous by nature. Annotating a sarcastic corpus has been a long standing problem, with sarcastic comments representing $< 1\%$ of all text on social media (Reddit, for example). This renders it infeasible to blindly annotate texts since finding an instance of sarcasm is like searching for a needle in a haystack. Consequently, papers have traditionally relied on various heuristics to generate these datasets—like using the self-disclosed /s tag or asking users to share their own sarcastic Tweets (our task). These heuristics, however, lead to noisy labels and annotator bias [17].

Generating synthetic responses. To generate the synthetic data, we used ChatGPT.[1] The generation parameters for the model were set to `temperature: 1`, `top p: 1`, `frequency penalty: 0.5`, `presence penalty: 0.4`, `max tokens: 700`. We chose these parameters to maximize the diversity in the decoded text. The frequency penalty reduces the probability of a word depending on the frequency that it occurs, while the presence penalty puts a flat cost on each word when it occurs in the text. These two forms of penalties help encourage the model to produce higher perplexity texts instead of selecting the next most probable word.

The generative data is then processed by removing artifacts of the generation. We defined these rules based on manual examination. The two most common problems that occurred were the model responding to the request in the affirmative ("Sure, here you go:") and outlining which taxonomy it uses prior to generating the sentence (only present in the taxonomy generation prompting). Both of these issues were addressed by splitting the first colon character ":" and restricting to text after it.

Model finetuning. Similar to previous work, we fine-tune a `E5-base` model on the synthetic data [18], [19]. This model was originally trained using a contrastive loss and achieves strong performance in a fine-tuned classification setting.

We finetune our model across two different settings: one-step (baseline), two-step (SAM). In the baseline setting we use a AdamW. In the SAM-setting we finetuned using the AdamW as the base optimizer and the two additional training steps (refer above). The code for the SAM optimizer was taken from an online repository, available here: https://github.com/davda54/sam.

We trained on 2000 datapoints across all four settings since this is typically enough data points to finetune a pretrained model [20]. To document our runs we used Weights and Biases to keep track of runs and performance.

[1] https://openai.com/blog/chatgpt

Modelling approach. Our modelling approach is centered on quantifying the potential of an advanced optimizer to counteract the distribution shift from synthetic data. We investigate this through four distinct training setups: real data alone, real data with Sharpness Aware Minimization (SAM), synthetic data alone, and synthetic data with SAM. Utilizing each of these configurations, we finetune a model focused on the sarcasm task. Following this, an evaluation is conducted for each scenario to scrutinize their relative performance. Explicitly:

$$\textit{SAM-value} = (F1_{ss} - F1_{sn}) - (F1_{rs} - F1_{rn})$$

Where $F1$ refers to the macro-F1 score and the subscripts indicate synthetic ($s*$) or real ($r*$) and with SAM ($*s$) or without SAM ($*n$). In words, this captures the relative gain of training with SAM for the synthetic setting than training without SAM.

IV. Results and Evaluation

In this section we present the results from fine-tuning the model with SAM on the synthetic and real data. We begin by discussing our approach for hyperparameter search and then get into the results of the different models.

Hyper-parameter search. In the setting without SAM we stuck to a learning rate of $2e^{-5}$ as done in previous work [18], [3]. For the setting with SAM we used grid search over the learning rate and ρ (SAM step) to find the optimal set of hyperparameters. For the learning rate we searched over a log uniform space between $[1e^{-3}, 1e^{-6}]$ and for the step we used $[0.001, 0.2]$ (corresponding to previously used steps). Since hyperparameter search is expensive, we utilized a smaller sample of our main dataset to find the optimal set of hyperparameters. Specifically, we limited to 640 data points which is over a quarter of the original dataset and conducted parameter search there over 1 epoch. In the end, the best model had a learning rate of $2e^{-5}$ and a SAM step of 0.003.

Furthermore, we trained for 3 epochs. This number was chosen by early-stopping: after it the validation loss saturates.

Results. In Table I we report the performance across our four training setups we tried. We first notice that training with SAM is strictly worse than training without it. For us this was a surprise since the related work has traditionally suggested the SAM increases generalizability. In general, we find that there is a drop in Macro-F1 across the two settings where training with SAM in the real setting drops the macro-F1 by one point, whereas training with it in the synthetic setup drops the macro-f1 by three points. Coming back to our SAM score (see above), we calculate it to be -0.02.

Approach	*Sarcasm*		
	Accuracy	Macro-F1	Precision
Real	0.67	0.50	0.51
Real + SAM	0.64	0.49	0.49
Synthetic	0.31	0.29	0.63
Synthetic + SAM	0.29	0.26	0.63

TABLE I
PERFORMANCE ACROSS DIFFERENT SETUPS ON THE SARCASM DATASET.

V. Discussion & Conclusion

In this paper, we focused on leveraging sharpness-aware minimization to improve the performance of LLMs fine-tuned with synthetic data. Using synthetic data in machine learning may be beneficial, especially in situations where acquiring real data is expensive and time-consuming. However, overfitting or biases create challenges in utilizing it. Therefore, we attempted to mitigate these issues by targeting flatter loss minima with SAM to improve model generalization.

While our results did not yield as significant improvement as originally hypothesized, they do contribute insights into the interplay between synthetic data, optimization methods and LLMs. The slight increase in accuracy that can be seen when using SAM with synthetic data indicates that the choice of optimization method can play a role in overcoming the limitations of synthetic data. A clear limitation of our approach is the fact that it is only tested on one specific textual construct - text instances of sarcasm. For other constructs the impact of using SAM can vary and should be studied separately. Moreover, with little variance in our model architecture, the model might have its own inductive bias, that could negatively influence its performance on unseen data.

For the next steps, a more rigorous exploration of the model parameters could be conducted, especially by extending hyperparameters tuning to all four variants and not only setups with synthetic data. This optimization step was done using grid search, to extensively explore the chosen parameter space, but could be also performed using random search or Bayesian optimization. These methods often provide better efficiency in parameter optimization, however in the case of fine-tuning pre-trained language models, they can fail to outperform grid search [21]. Nevertheless it remains a possible improvement aspect for further investigation. To gain a better insight into the impact of our optimization methods and network architecture, visualizing the loss surface would be an interesting step for the future. An approach for this could be inspired by [22] where they use filter-wise normalized directions to visualize the curvature of a loss function and moreover perform comparative analyses between different loss functions. The resulting visualizations can then explore the sharpness of minima found during training. However, generating those loss surface plots comes with high computational complexity, particularly since we are working with large-scale language models.

Synthetic data generation is not going anywhere; in fact, as LLMs get larger and run out of human data, they will require more data to train them. One source for this data will be synthetically generated text to make it meaningful to humans. For this reason it's critically important to develop a method to successfully extract signal from this synthetic text. In this paper we explored one such method aiming at improving the optimization process.

REFERENCES

[1] R. Tang, X. Han, X. Jiang, and X. Hu, "Does synthetic data generation of llms help clinical text mining?" 2023.

[2] X. He, I. Nassar, J. Kiros, G. Haffari, and M. Norouzi, "Generate, Annotate, and Learn: NLP with Synthetic Text," *Transactions of the Association for Computational Linguistics*, vol. 10, pp. 826–842, 08 2022. [Online]. Available: https://doi.org/10.1162/tacl_a_00492

[3] V. Veselovsky, M. H. Ribeiro, A. Arora, M. Josifoski, A. Anderson, and R. West, "Generating faithful synthetic data with large language models: A case study in computational social science," 2023.

[4] D. Bahri, H. Mobahi, and Y. Tay, "Sharpness-aware minimization improves language model generalization," 2022.

[5] P. Foret, A. Kleiner, H. Mobahi, and B. Neyshabur, "Sharpness-aware minimization for efficiently improving generalization," 2021.

[6] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang, "On large-batch training for deep learning: Generalization gap and sharp minima," 2017.

[7] Y. Jiang, B. Neyshabur, H. Mobahi, D. Krishnan, and S. Bengio, "Fantastic generalization measures and where to find them," 2019.

[8] M. Andriushchenko and N. Flammarion, "Towards understanding sharpness-aware minimization," 2022.

[9] J. Kwon, J. Kim, H. Park, and I. K. Choi, "Asam: Adaptive sharpness-aware minimization for scale-invariant learning of deep neural networks," 2021.

[10] Y. Papanikolaou and A. Pierleoni, "Dare: Data augmented relation extraction with gpt-2," *arXiv preprint arXiv:2004.13845*, 2020.

[11] A. Abaskohi, A. Rasouli, T. Zeraati, and B. Bahrak, "Utnlp at semeval-2022 task 6: A comparative analysis of sarcasm detection using generative-based and mutation-based data augmentation," *arXiv preprint arXiv:2204.08198*, 2022.

[12] R. Sennrich, B. Haddow, and A. Birch, "Improving neural machine translation models with monolingual data," *arXiv preprint arXiv:1511.06709*, 2015.

[13] U. Maqsud, "Synthetic text generation for sentiment analysis," in *Proceedings of the 6th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, 2015, pp. 156–161.

[14] R. Eldan and Y. Li, "Tinystories: How small can language models be and still speak coherent english?" *arXiv preprint arXiv:2305.07759*, 2023.

[15] M. Josifoski, M. Sakota, M. Peyrard, and R. West, "Exploiting asymmetry for synthetic training data generation: Synthie and the case of information extraction," *arXiv preprint arXiv:2303.04132*, 2023.

[16] I. A. Farha, S. V. Oprea, S. Wilson, and W. Magdy, "Semeval-2022 task 6: isarcasmeval, intended sarcasm detection in english and arabic," in *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, 2022, pp. 802–814.

[17] S. Oprea and W. Magdy, "isarcasm: A dataset of intended sarcasm," *arXiv preprint arXiv:1911.03123*, 2019.

[18] A. G. Møller, J. A. Dalsgaard, A. Pera, and L. M. Aiello, "Is a prompt and a few samples all you need? using gpt-4 for data augmentation in low-resource classification tasks," *arXiv preprint arXiv:2304.13861*, 2023.

[19] L. Wang, N. Yang, X. Huang, B. Jiao, L. Yang, D. Jiang, R. Majumder, and F. Wei, "Text embeddings by weakly-supervised contrastive pre-training," *arXiv preprint arXiv:2212.03533*, 2022.

[20] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2019.

[21] X. Liu and C. Wang, "An empirical study on hyperparameter optimization for fine-tuning pre-trained language models," 2021.

[22] H. Li, Z. Xu, G. Taylor, C. Studer, and T. Goldstein, "Visualizing the loss landscape of neural nets," in *Neural Information Processing Systems*, 2018.

APPENDIX

Figure 1 displays the model setup.

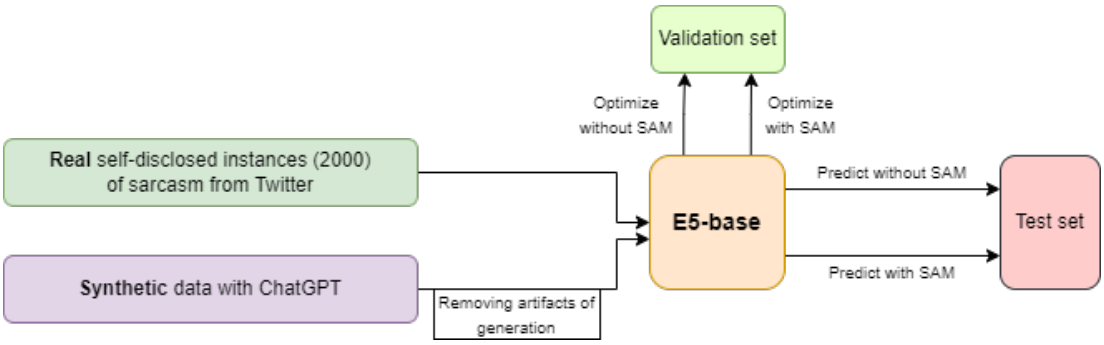

Fig. 1. Model setup diagram