Ensembling Finetuned Language Models for Text Classification

Sebastian Pineda Arango[1*], **Maciej Janowski**[1*], **Lennart Purucker**[1], **Arber Zela**[1], **Frank Hutter**[3,1], **Josif Grabocka**[2]

[1] University of Freiburg, [2] University of Technology Nürnberg, [3] ELLIS Institute Tübingen

Abstract

Finetuning is a common practice widespread across different communities to adapt pretrained models to particular tasks. Text classification is one of these tasks for which many pretrained models are available. On the other hand, ensembles of neural networks are typically used to boost performance and provide reliable uncertainty estimates. However, ensembling pretrained models for text classification is not a well-studied avenue. In this paper, we present a metadataset with predictions from five large finetuned models on six datasets, and report results of different ensembling strategies from these predictions. Our results shed light on how ensembling can improve the performance of finetuned text classifiers and incentivize future adoption of ensembles in such tasks.

1 Introduction

In recent years, fine-tuning pretrained models has become a widely adopted technique for adapting general-purpose models to specific tasks (Arango et al., 2023). This practice has gained significant traction across various communities due to its effectiveness in leveraging the vast knowledge encoded in pretrained models. Among the diverse tasks that benefit from fine-tuning, text classification stands out as one of the most prevalent. With the availability of numerous pretrained models, practitioners often find themselves with a range of powerful tools to tackle text classification challenges. However, despite the widespread use of fine-tuning, the potential benefits of combining or ensembling these fine-tuned models remain underexplored.

Previous studies have primarily concentrated on improving individual model performance through fine-tuning techniques (Howard & Ruder, 2018), leaving the exploration of ensemble strategies largely underdeveloped in this context. This oversight is particularly significant given the well-documented advantages of model ensembling in other machine learning domains (Erickson et al., 2020; Lakshminarayanan et al., 2017), which has been shown to enhance robustness and generalization. In this paper, we address the aforementioned gap by introducing a novel metadataset, which we dub: Finetuning Text Classifiers (FTC) metadataset. FTC contains predictions from various fine-tuned models on text classification tasks with various number of classes. We systematically evaluate different ensembling strategies using this metadataset, aiming to uncover insights into the potential improvements that ensembling can offer. Our results provide valuable evidence on the efficacy of these strategies, demonstrating that ensembling fine-tuned models can lead to performance gains in text classification.

2 Background and Related Work

Finetuning for Text Classification. Universal Language Model Fine-tuning for Text Classification or ULMFiT (Howard & Ruder, 2018) consists of finetuning language models for classification in two

[*]Equal contribution. Corresponding author: pineda@cs.uni-freiburg.de

38th Workshop on Fine-Tuning in Machine Learning (NeurIPS 2024).

stages: 1) a target task unsupervised finetuning and 2) target task classifier finetuning, while using a different learning rate per layer. However, the feasibility of fully fine-tuning large pretrained language models is constrained by computational limits (Radford et al., 2018). This has spurred the adoption of Parameter-Efficient Fine-Tuning (PEFT) methods (Han et al., 2024). Early strategies focused on minimal subsets of parameters such as sparse subnetworks (Sung et al., 2021) to improve task-specific performance efficiently. Innovations such as adapter modules (Houlsby et al., 2019), which introduce a few parameters per transformer layer but in consequence increase inference time, prompted the development of Low-Rank Adaptation (LoRA) (Hu et al., 2022; Dettmers et al., 2024) that applies low-rank updates for improved downstream task performance with reduced computational overhead. Some studies have also demonstrated that finetuned language models can be ensembled to improve performance for text classification (Abburi et al., 2023), but they do not provide clear insights about ensembling methods, hyperparameters, or metadata.

Ensembling Deep Learning Models. Ensembles of neural networks (Hansen & Salamon, 1990; Krogh & Vedelsby, 1995; Dietterich, 2000) have gained significant attention in deep learning research, both for their performance-boosting capabilities and their effectiveness in uncertainty estimation. Various strategies for building ensembles exist, with deep ensembles (Lakshminarayanan et al., 2017) being the most popular one, which involve independently training multiple initializations of the same network. Their state-of-the-art predictive uncertainty estimates have further fueled the interest in ensembles. Extensive empirical studies (Ovadia et al., 2019; Gustafsson et al., 2020) have shown that deep ensembles outperform other approaches for uncertainty estimation, such as Bayesian neural networks (Blundell et al., 2015; Gal & Ghahramani, 2016; Welling & Teh, 2011). Similar to our work, Seligmann et al. (2024) show that finetuning pretrained models via Bayesian methods on the WILDS dataset (Koh et al., 2021), which contains text classification as well, can yield significant performance as compared to standard finetuning of single models.

Post-Hoc Ensembling (PHE). PHE uses set of fitted base models $\{z_1, ..., z_M\}$ such that every model outputs $z_m(x), z_m : \mathbb{R}^D \rightarrow \mathbb{R}^C$ [2]. These outputs are combined by an ensembler $f(z_1(x), ..., z_M(x); \theta) = f(z(x); \theta)$, where $z(x) = [z_1(x), ..., z_M(x)]$ is the concatenation of the base models predictions. While the base models learned from a training set $\mathcal{D}_{\text{Train}}$, the ensembler's parameters θ are typically obtained by minimizing a loss function $\mathcal{L}$ on a validation set $\mathcal{D}_{\text{Val}}$ such that:

$$\theta \in \arg\min_{\theta} \sum_{(x,y)\in\mathcal{D}_{\text{Val}}} \mathcal{L}(f(z(x), y; \theta)). \tag{1}$$

A popular approach is a linear combination of the model outputs as $f(z(x); \theta) = \sum_m \theta_m z_m(x)$.

PHE Metadatasets. Similarly, prior studies have created metadatasets containing the *predictions* of base models, but only for time-series (Borchert et al., 2022) and tabular (Purucker & Beel, 2022, 2023; Purucker et al., 2023; Salinas & Erickson, 2023) data.

3 Finetuning Text Classifiers (FTC) Metadataset

Search Space. Our search space comprises three hyperparameters: the model type, learning rate and LoRA rank (Hu et al., 2022). We consider five model choices: 1) **GPT2, 124M** parameters; (Radford et al., 2019); 2) **Bert-Large, 336M** ; (Devlin et al., 2018); 3) **Bart-Large, 400 M**, parameters (Lewis et al., 2019); 4) **Albert-Large, 17M** parameters (Lan et al., 2019); and 5) **T5-Large, 770 M** parameters (Raffel et al., 2020). For the other two hyperparameters we also consider five different discrete values as specified in Table 1.

Table 1: Search Space parameterization.

Hyperparameter	**Values**
Model	GPT2, Bert-Large, Albert-Large, Bart-Large, T5-Large
Learning Rate	0.00001, 0.0001, 0.0005, 0.001, 0.005
LoRA Rank	8, 16, 32, 64, 128

Datasets. The metadataset contains predictions of models finetuned on five metadatasets for text classification: 1) *IMDB* (Maas et al., 2011); 2) *Tweet* (Maggie, 2020), 3) *News* (Zhang et al., 2015), 4) *DBpedia* (Zhang et al., 2015), 5) *SST2* (Socher et al., 2013) and 6) *SetFit* (Tunstall et al., 2021). We created two versions for every dataset: the first is trained with the complete training data, while the

[2] We assume a classification tasks with C classes. For regression $C = 1$.

Table 2: Metadataset information.

Dataset	# Classes	# Train Samples	# Val. Samples	# Test Samples	# Confs (100%)	# Confs. (10%)
IMDB (Maas et al., 2011)	2	20,000	5,000	25,000	125	125
Tweet (Maggie, 2020)	3	27,485	5,497	3,534	100	100
News (Zhang et al., 2015)	4	96,000	24,000	7,600	99	120
DBpedia (Zhang et al., 2015)	14	448,000	112,000	70,000	25	65
SST-2 (Socher et al., 2013)	2	43,103	13,470	10,776	125	125
SetFit (Tunstall et al., 2021)	3	393,116	78,541	62,833	25	100

second is only with a subset of 10% of the samples. All the datasets are for text classification from 2 to 14 classes, including diverse domains such as movies, reviews, news, tweets, and text entailment data. We provide further information on the datasets in Table 2 and Appendix A.

Metadataset Creation and Composition.[3] We created the dataset by finetuning every model to the train split and, subsequently, saving their predictions on the validation and test split. This allows us to quickly simulate ensembling methods given the precomputed predictions. The validation split corresponds to 20% of the available train data. For *SST-2* and *SetFit* the test data is not completely provided by the creators, or it has hidden labels, therefore, we obtain it by using 20% of the remaining training data. The models are finetuned up to 5 epochs using a single Nvidia A100 GPU with batch size set to 2 and no LoRA dropout. We vary only the model type, learning rate, and LoRA rank, while keeping the other hyperparameters to their default values in the TRAINER object from the *Transformers Library* (v4.41.0) [4]. In total, the metadataset contains 1134 evaluated configurations, representing around 3800 GPU hours of computation. Additionally, we report information about the metadataset in Table 2, as well as training times per dataset in Table 6.

Table 3: Best configuration per dataset.

	100 %			10 %		
Dataset	Model	Learning Rate	Lora Rank	Model	Learning Rate	Lora Rank
DBpedia	GPT2	0.0001	64	Bert	0.0001	16
News	Bart	0.0001	64	Bart	0.0001	128
SetFit	GPT2	0.0001	128	Bart	0.0001	8
SST2	T5	0.0001	8	T5	0.0001	64
Tweet	Bart	0.0001	64	Bart	0.0001	64
IMDB	Bart	0.0001	128	T5	0.0001	64

Hyperparameter Importance. We explore the importance of two hyperparameters, learning rate, and LoRA rank, by plotting the mean error as a heatmap in Figure 1. The error corresponds to the average across different models and datasets. We can notice that the learning rate is an important hyperparameter, while increasing the LoRA rank does not affect the performance significantly in the low learning rate regime. This behaviour is interesting, as it showcases that a small rank is enough for successful finetuning in this context. A similar pattern arises when using 10% of the data, as shown in Figure 2 in the appendix. To compare the different classifiers, we report their test error on all dataset versions after selecting the best LoRA rank and learning rate configuration, in Table 7. T5-large shows strong performance in comparison to the other models. Bart and GPT2 also outperform the rest of the models in some datasets. These results demonstrate that the model type is also a relevant hyperparameter, which might motivate the exploration of joint model/architecture and hyperparameter optimization for achieving the best performance.

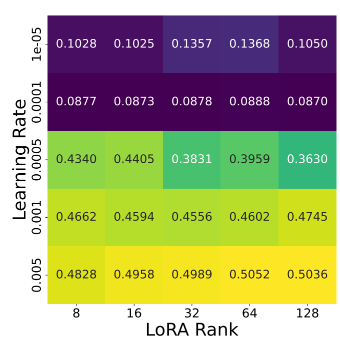

Figure 1: Mean error across datasets for different hyperparameter combinations.

4 Benchmarking Ensembles of Finetuned Text Classifiers

We compare the Neural Ensemblers with other common and competitive ensemble approaches. 1) **Single best** selects the best model according to the validation metric; 2) **Random-N** chooses randomly N models to ensemble, 3) **Top-N** ensembles the best N models according to the validation metric; 4) **Greedy-N** creates an ensemble with N models by iterative selecting the one that improves the

[3]Access to the metadataset and finetuning code in `https://github.com/sebastianpinedaar/finetuning_text_classifiers`

[4]Although we evaluate the models in a grid, some runs yielded out-of-memory errors for some configurations.

Table 4: Classification error per dataset.

Method	DBpedia		News		SetFit		SST-2		Tweet		IMDB	
	100 %	10 %	100 %	10 %	100 %	10 %	100 %	10 %	100 %	10 %	100 %	10 %
Single-Best	0.0077	0.0085	0.0462	0.0657	0.1898	0.1338	0.0396	0.0507	0.2012	0.2306	0.0362	0.0455
Random-5	0.0139	0.3157	0.0574	0.0833	0.2383	0.1624	0.0542	0.1060	0.1925	0.2233	0.0507	0.0657
Random-50	0.0110	0.0082	0.0558	0.0786	0.1965	0.1639	0.0529	0.0684	0.1898	0.2140	0.0387	0.0497
Top-5	0.0076	0.0077	**0.0455**	0.0636	0.1846	0.1277	**0.0359**	0.0488	0.1921	0.2187	0.0328	**0.0416**
Top-50	0.0110	0.0083	0.0525	0.0651	0.1989	0.1526	0.0411	0.0543	0.1885	0.2142	0.0370	0.0446
Model Average	0.0087	0.0087	0.0533	0.0703	0.1896	0.1450	0.0444	0.0564	0.1889	0.2107	0.0392	0.0484
Greedy-5	**0.0074**	0.0079	0.0459	0.0611	0.1846	0.1261	0.0377	**0.0472**	0.1953	0.2102	**0.0321**	0.0420
Greedy-50	0.0075	**0.0076**	0.0459	**0.0593**	**0.1843**	**0.1245**	0.0376	0.0473	**0.1872**	**0.2050**	**0.0321**	0.0420

Table 5: Negative log-likelihood (NLL) per dataset.

Method	DBpedia		News		SetFit		SST-2		Tweet		IMDB	
	100 %	10 %	100 %	10 %	100 %	10 %	100 %	10 %	100 %	10 %	100 %	10 %
Single-Best	0.0497	0.0631	0.2085	0.3369	0.8154	0.6112	0.2037	0.2186	0.6225	0.7870	0.1475	0.2086
Random-5	0.0644	1.0705	0.5032	0.4500	0.8871	0.6488	0.2959	0.4100	0.6763	0.6745	0.2856	0.3051
Random-50	0.0492	0.3900	0.5706	0.4091	0.6728	0.6434	0.3447	0.3788	0.6466	0.6939	0.3483	0.3551
Top-5	0.0424	0.0534	0.1768	0.2423	0.7175	0.4945	0.1468	0.2159	0.5822	0.7060	0.1193	0.1576
Top-50	0.0484	0.2355	0.1796	0.2348	0.6997	0.6379	0.1275	0.2034	0.5181	0.7223	0.1179	0.1320
Model Average	0.0433	0.1453	0.2461	0.2753	0.5541	0.4602	0.1685	0.1987	0.5143	0.5588	0.1561	0.1716
Greedy-5	0.0383	0.0446	0.1751	0.2319	0.5413	0.4037	0.1389	0.1587	0.5085	0.5419	0.1150	0.1272
Greedy-50	**0.0358**	**0.0364**	**0.1582**	**0.1978**	**0.5290**	**0.3572**	**0.1167**	**0.1365**	**0.4769**	**0.5077**	**0.1031**	**0.1241**

metric as proposed by previous work (Caruana et al., 2004, 2006); 5) **Model Average (MA)** simply computes the sum of the predictions with constant weights. For some baselines, we tried both 5 and 50 models in the ensembles, e.g. *Greedy-50* has 50 models.

4.1 Observation 1: Ensembling finetuned text classifiers is helpful.

To understand whether it is helpful to ensemble finetunined text classifier, we evaluate the baselines on the six datasets, on both versions with 100% and 10% of the training data. We measure the negative log-likelihood (NLL) and the classification error on the test data, while we use the validation split for training the ensemble. From results shown in Tables 4 and 5, we observe that the best method (bold-faced) is always an ensembling technique. Except Random-N, all the other ensembling strategies yield consistently better results than the single-best approach, which corresponds to a grid search on the search space of configurations. Particularly, we notice that the Greedy-N approach is very strong across all datasets, especially regarding the NLL. A large ensemble (50 base models) seems to be beneficial using the *Greedy-N* approach, but the results are mixed when using the *Top-50* or *Random-50*. We notice a particular large improvement in the NLL metric, confirming that ensembling provides robustness and better uncertainty calibrations (Lakshminarayanan et al., 2017; Ovadia et al., 2019; Seligmann et al., 2024).

4.2 Observation 2: Ensembling text classifiers finetuned on 10% of the training data yields strong results.

Given the two training splits in the metadataset, we study the advantages of using just 10% of the data for finetuning and post-hoc ensembling. Our results show that, as expected, the best option is to use the whole training data. Nevertheless, we notice that ensembling is also beneficial when training in the subset of data (see Tables 4 and 5). Remarkably, ensembling these models sometimes yields better performance than using a single best trained on the whole data. We observe such results for all models under NLL and for two models using *Greedy-50* under the error metrics.

5 Conclusion

In this work, we introduced a metadataset containing the predictions of finetuned text classifiers and evaluated common ensembling strategies using this data. Our study provided insights on how simple strategies can improve on top of vanilla single configuration selection in the context of text classification. We empirically showed that even finetuning on small datasets or subsets of data can yield a considerable improvement. Finally, our experiments suggest that the finetuned model and learning rate have an important impact on the final performance.

Acknowledgments

The authors gratefully acknowledge the scientific support and HPC resources provided by the Erlangen National High Performance Computing Center (NHR@FAU) of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) under the NHR project v101be. NHR funding is provided by federal and Bavarian state authorities. NHR@FAU hardware is partially funded by the German Research Foundation (DFG) – 440719683. This research was partially supported by the following sources: TAILOR, a project funded by EU Horizon 2020 research and innovation programme under GA No 952215; the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under grant number 417962828 and 499552394 - SFB 1597; the European Research Council (ERC) Consolidator Grant "Deep Learning 2.0" (grant no. 101045765). Frank Hutter acknowledges financial support by the Hector Foundation. The authors acknowledge support from ELLIS and ELIZA. Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the ERC. Neither the European Union nor the ERC can be held responsible for them.

Funded by
the European Union

.

References

Abburi, H., Suesserman, M., Pudota, N., Veeramani, B., Bowen, E., and Bhattacharya, S. Generative AI text classification using ensemble LLM approaches. In Montes-y-Gómez, M., Rangel, F., Jiménez-Zafra, S. M., Casavantes, M., Altuna, B., Álvarez-Carmona, M. Á., Bel-Enguix, G., Chiruzzo, L., de la Iglesia, I., Escalante, H. J., Cumbreras, M. Á. G., García-Díaz, J. A., Barba, J. Á. G., Tamayo, R. L., Lima, S., Moral, P., del Arco, F. M. P., and Valencia-García, R. (eds.), *Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2023) co-located with the Conference of the Spanish Society for Natural Language Processing (SEPLN 2023), Jaén, Spain, September 26, 2023*, volume 3496 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2023.

Arango, S. P., Ferreira, F., Kadra, A., Hutter, F., and Grabocka, J. Quick-tune: Quickly learning which pretrained model to finetune and how. *arXiv preprint arXiv:2306.03828*, 2023.

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. Weight uncertainty in neural network. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1613–1622, Lille, France, 07–09 Jul 2015. PMLR. URL `http://proceedings.mlr.press/v37/blundell15.html`.

Borchert, O., Salinas, D., Flunkert, V., Januschowski, T., and Günnemann, S. Multi-objective model selection for time series forecasting. *arXiv preprint arXiv:2202.08485*, 2022.

Caruana, R., Niculescu-Mizil, A., Crew, G., and Ksikes, A. Ensemble selection from libraries of models. In *Proceedings of the twenty-first international conference on Machine learning*, pp. 18, 2004.

Caruana, R., Munson, A., and Niculescu-Mizil, A. Getting the most out of ensemble selection. In *Sixth International Conference on Data Mining (ICDM'06)*, pp. 828–833. IEEE, 2006.

Dettmers, T., Pagnoni, A., Holtzman, A., and Zettlemoyer, L. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36, 2024.

Devlin, J., Chang, M., Lee, K., and Toutanova, K. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL `http://arxiv.org/abs/1810.04805`.

Dietterich, T. G. Ensemble Methods in Machine Learning. In *Multiple Classifier Systems*, pp. 1–15, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg. ISBN 978-3-540-45014-6.

Erickson, N., Mueller, J., Shirkov, A., Zhang, H., Larroy, P., Li, M., and Smola, A. Autogluon-tabular: Robust and accurate automl for structured data. *arXiv preprint arXiv:2003.06505*, 2020.

Gal, Y. and Ghahramani, Z. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pp. 1050–1059, New York, New York, USA, 20–22 Jun 2016. PMLR. URL `http://proceedings.mlr.press/v48/gal16.html`.

Gustafsson, F. K., Danelljan, M., and Schön, T. B. Evaluating Scalable Bayesian Deep Learning Methods for Robust Computer Vision. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2020.

Han, Z., Gao, C., Liu, J., Zhang, J., and Zhang, S. Q. Parameter-efficient fine-tuning for large models: A comprehensive survey, 2024. URL `https://arxiv.org/abs/2403.14608`.

Hansen, L. K. and Salamon, P. Neural network ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(10):993–1001, 1990.

Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., De Laroussilhe, Q., Gesmundo, A., Attariyan, M., and Gelly, S. Parameter-efficient transfer learning for NLP. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2790–2799. PMLR, 09–15 Jun 2019. URL `https://proceedings.mlr.press/v97/houlsby19a.html`.

Howard, J. and Ruder, S. Universal language model fine-tuning for text classification. In Gurevych, I. and Miyao, Y. (eds.), *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pp. 328–339. Association for Computational Linguistics, 2018.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=nZeVKeeFYf9`.

Koh, P. W., Sagawa, S., Marklund, H., Xie, S. M., Zhang, M., Balsubramani, A., Hu, W., Yasunaga, M., Phillips, R. L., Gao, I., Lee, T., David, E., Stavness, I., Guo, W., Earnshaw, B., Haque, I., Beery, S. M., Leskovec, J., Kundaje, A., Pierson, E., Levine, S., Finn, C., and Liang, P. Wilds: A benchmark of in-the-wild distribution shifts. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 5637–5664. PMLR, 18–24 Jul 2021.

Krogh, A. and Vedelsby, J. Neural Network Ensembles, Cross Validation, and Active Learning. In *Advances in Neural Information Processing Systems 7*, pp. 231–238. MIT Press, 1995.

Lakshminarayanan, B., Pritzel, A., and Blundell, C. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., and Soricut, R. ALBERT: A lite BERT for self-supervised learning of language representations. *CoRR*, abs/1909.11942, 2019. URL `http://arxiv.org/abs/1909.11942`.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *CoRR*, abs/1910.13461, 2019. URL `http://arxiv.org/abs/1910.13461`.

Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., and Potts, C. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 142–150, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/P11-1015`.

Maggie, Phil Culliton, W. C. Tweet sentiment extraction, 2020. URL `https://kaggle.com/competitions/tweet-sentiment-extraction`.

Mendes, P. N., Jakob, M., and Bizer, C. *DBpedia: A multilingual cross-domain knowledge base*. European Language Resources Association (ELRA), 2012.

Nangia, N., Williams, A., Lazaridou, A., and Bowman, S. R. The repeval 2017 shared task: Multi-genre natural language inference with sentence representations. *arXiv preprint arXiv:1707.08172*, 2017.

Ovadia, Y., Fertig, E., Ren, J., Nado, Z., Sculley, D., Nowozin, S., Dillon, J., Lakshminarayanan, B., and Snoek, J. Can you trust your model's uncertainty? Evaluating predictive uncertainty under dataset shift. In *Advances in Neural Information Processing Systems 32*, pp. 13991–14002. Curran Associates, Inc., 2019.

Purucker, L. O. and Beel, J. Assembled-openml: Creating efficient benchmarks for ensembles in automl with openml. In *First Conference on Automated Machine Learning (Late-Breaking Workshop)*, 2022.

Purucker, L. O. and Beel, J. Cma-es for post hoc ensembling in automl: A great success and salvageable failure. In *International Conference on Automated Machine Learning*, pp. 1–1. PMLR, 2023.

Purucker, L. O., Schneider, L., Anastacio, M., Beel, J., Bischl, B., and Hoos, H. Q(d)o-es: Population-based quality (diversity) optimisation for post hoc ensemble selection in automl. In *International Conference on Automated Machine Learning*, pp. 10–1. PMLR, 2023.

Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. Improving language understanding with unsupervised learning. Technical Report, OpenAI, 2018. URL `https://openai.com/index/language-unsupervised/`.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language models are unsupervised multitask learners. 2019.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. URL `http://jmlr.org/papers/v21/20-074.html`.

Salinas, D. and Erickson, N. Tabrepo: A large scale repository of tabular model evaluations and its automl applications. *arXiv preprint arXiv:2311.02971*, 2023.

Seligmann, F., Becker, P., Volpp, M., and Neumann, G. Beyond deep ensembles: a large-scale evaluation of bayesian deep learning under distribution shift. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NeurIPS '23, Red Hook, NY, USA, 2024. Curran Associates Inc.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A., and Potts, C. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1631–1642, Seattle, Washington, USA, October 2013. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/D13-1170`.

Sung, Y.-L., Nair, V., and Raffel, C. A. Training neural networks with fixed sparse masks. *Advances in Neural Information Processing Systems*, 34:24193–24205, 2021.

Tunstall, L., Pereg, O., Bates, L., Wasserblat, M., Eun, U., Korat, D., Reimers, N., and Aarsen, T. Setfit-mnli, 2021. URL `https://huggingface.co/datasets/SetFit/mnli`.

Tunstall, L., Reimers, N., Jo, U. E. S., Bates, L., Korat, D., Wasserblat, M., and Pereg, O. Efficient few-shot learning without prompts, 2022. URL `https://arxiv.org/abs/2209.11055`.

Welling, M. and Teh, Y. W. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pp. 681–688, Madison, WI, USA, 2011. Omnipress. ISBN 9781450306195.

Zhang, X., Zhao, J., and LeCun, Y. Character-level convolutional networks for text classification. In Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. URL `https://proceedings.neurips.cc/paper_files/paper/2015/file/250cf8b51c773f3f8dc8b4be867a9a02-Paper.pdf`.

A Details on Datasets

IMDB (Maas et al., 2011) The IMDB dataset contains reviews for movies and their binary sentiment. We only use the labeled training and test data. The data source we used is `https://huggingface.co/datasets/stanfordnlp/imdb`.

Tweet (Maggie, 2020) The Tweet dataset contains the text of tweets and their sentiment label. The data was initially curated for a Kaggle competition. The data source we used is `https://www.kaggle.com/competitions/tweet-sentiment-extraction`.

DBpedia and News (Zhang et al., 2015) The DBpedia and News datasets were created by Zhang et al. (2015) for benchmarking deep learning models for text classification tasks.

We use the AG's News dataset, consisting of the title and description fields of news articles from the web. The data source we used is `https://huggingface.co/datasets/fancyzhx/ag_news`.

The DBpedia dataset contains the title and abstract of Wikipedia articles sourced from DBpedia 2014 (Mendes et al., 2012). The data source we used is `https://huggingface.co/datasets/fancyzhx/dbpedia_14`.

SST-2 (Socher et al., 2013) The Stanford Sentiment Treebank with two classes (SST-2) is a corpus of individual sentences from movie reviews. Three human judges labeled the sentences as having (somewhat) negative or (somewhat) positive sentiments. The data source we used is `https://huggingface.co/datasets/stanfordnlp/sst2`.

SetFit (Tunstall et al., 2021) Lastly, we use the SetFit (Tunstall et al., 2022) version of the Multi-Genre Natural Language Inference (MNLI) corpus (Nangia et al., 2017) as a dataset. The corpus encompasses text pairs from various sources, such as transcribed speech or fiction. Each text pair is labeled with whether one text entails the other, contradicts the other, or if they are neutral to each other. The data source we used is `https://huggingface.co/datasets/SetFit/mnli`.

B Additional Results

We present additional results:

- Mean error for different hyperparameters using a subset of data in Figure 2.
- Finetuning time for every dataset in Table 6.
- Comparison performance per model in Table 7.
- Comparison of different values of LoRA rank dimension in Figures 3

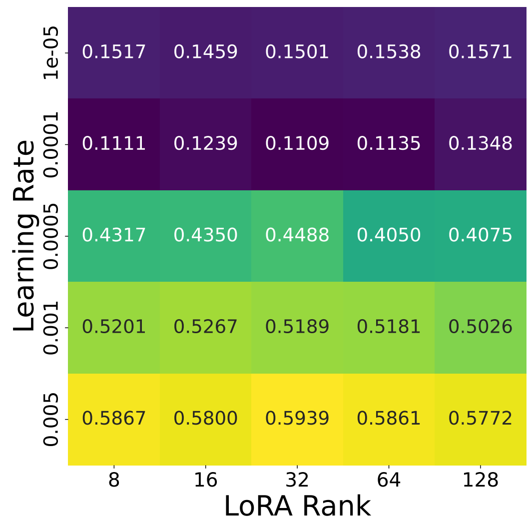

Figure 2: Error for different hyperparameters using 10 % of the data.

Table 6: Training times per dataset.

	Average (Min.)		**Total (Hrs.)**	
	Extended	**Mini**	**Extended**	**Mini**
Set-Fit	104.49	24.32	217.6963	405.4354
News	91.6443	12.20	756.0661	244.1131
DBPedia	186.22	36.99	387.9752	400.8265
IMDB	26.84	2.71	279.64	56.47
Tweet	34.54	3.46	287.83	57.77
SST2	57.97	5.79	603.94	120.63

Table 7: Error per Model.

Method	**IMDB**		**Tweet**		**News**		**DBpedia**		**SST2**		**Set-Fit**	
	100 %	**10 %**	**100 %**	**10 %**	**100 %**	**10 %**	**100 %**	**10 %**	**100 %**	**10 %**	**100 %**	**10 %**
GPT2	0.0576	0.0817	-	-	0.0611	0.0736	**0.0077**	0.0103	0.0840	0.1174	**0.1898**	0.2388
Bert-Large	0.0540	0.0752	0.2031	0.2365	0.0540	0.0772	-	**0.0085**	0.0516	0.0809	-	0.2007
Albert-Large	0.0534	0.0650	0.2043	0.2439	0.0553	0.0807	-	0.0105	0.0513	0.0917	-	0.1901
Bart-Large	**0.0342**	0.0459	0.2011	0.2306	**0.0461**	**0.0656**	-	-	0.0482	0.0654	-	**0.1337**
T5-Large	0.0362	**0.0455**	**0.1972**	**0.2303**	-	0.0735	-	-	**0.0396**	**0.0506**	-	-

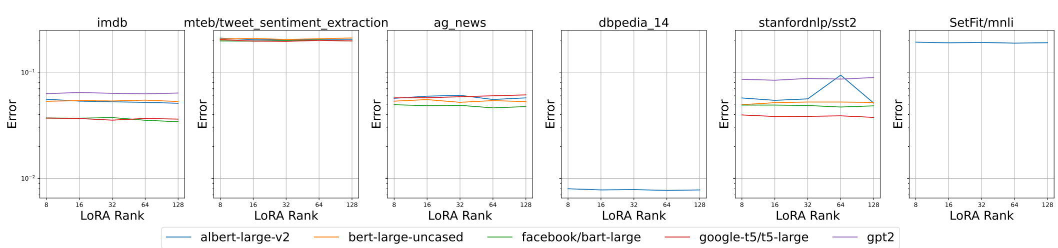

Figure 3: Error vs. LoRA Rank, *extended* version. The error variation is small across different LoRA rank values.

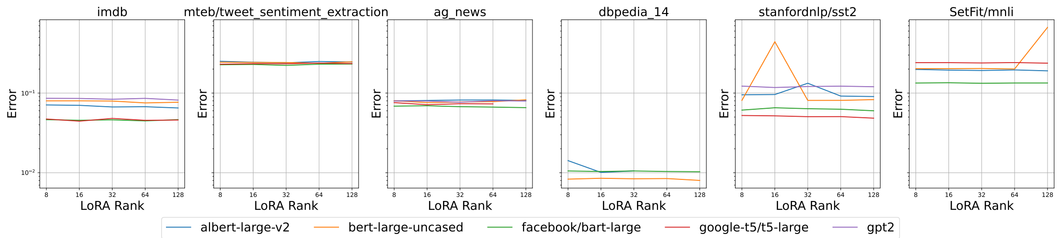

Figure 4: Error vs. LoRA Rank, *mini* version. The error variation is small across different LoRA rank values.