Enhancing BERT Fine-Tuning for Sentiment Analysis in Lower-Resourced Languages

Jozef Kubík and **Marek Šuppa** and **Martin Takáč**
Faculty of Mathematics Physics and Informatics
Comenius University in Bratislava, Slovakia

Abstract

Limited data for low-resource languages typically yield weaker language models (LMs). Since pre-training is compute-intensive, it is more pragmatic to target improvements during fine-tuning. In this work, we examine the use of Active Learning (AL) methods augmented by structured data selection strategies which we term 'Active Learning schedulers,' to boost the fine-tuning process with a limited amount of training data. We connect the AL to data clustering and propose an integrated fine-tuning pipeline that systematically combines AL, clustering, and dynamic data selection schedulers to enhance model's performance. Experiments in the Slovak, Maltese, Icelandic and Turkish languages show that the use of clustering during the fine-tuning phase together with AL scheduling can simultaneously produce annotation savings up to 30% and performance improvements up to four F1 score points, while also providing better fine-tuning stability.

1 Introduction

LMs such as BERT (Devlin, 2018) are considered breakthrough models in many areas. Although nowadays most of the research is done using prompting techniques on large foundational models (Shin et al., 2023), BERT family models are still used in lower-resource scenarios for a number of reasons. They allow full control[1] over fine-tuning, and their size and hardware requirements make them more easily feasible to use in the research and real-world environments. In contrast, using commercial models at scale can be expensive and often relies on Internet connection, meaning their use is not possible when the data in question cannot be shared with third parties (e.g. for privacy reasons).

There is a huge performance gap between high-resource languages, such as English, and lower-resourced languages in both prompting models and BERT-like models. However, this gap can be closed with clever modifications in both training phases, with the BERT family models being easier to train than GPT-like models (Achiam et al., 2023) due to their size and data requirements.

BERT's effectiveness often depends on large annotated datasets, highlighting the need for more efficient data exploration under limited annotation budgets. To address this, we propose a novel fine-tuning pipeline for lower-resourced language BERT models on the classification task. Our languages for the experiments representing lower-resourced languages will be Maltese, Icelandic, and Slovak, the last being the most popular of them spoken by approximately 5 million people (Short, 2018). To show possible generalization on more popular languages not considered low-resource but rather medium- to high-resource, we also conduct experiments on the Turkish language.

2 Related Work

The definition of low-resource can be interpreted in different ways depending on the language or setting (Nigatu et al., 2024). For the purpose of this study, we define low-resource operationally as a setting with severely limited annotated data.

Popular techniques in low-resource scenarios include adapting smaller models trained in English to other languages by modifying parts of the architecture (Al-Sallab et al., 2017) or transferring the knowledge of models pre-trained in high-resource languages to other models fine-tunable in low-resource languages (Heinzerling and Strube, 2017; Bojanowski et al., 2017). One of the alternative approaches, where the model itself helps identify the data samples which will help it to learn more efficiently, is called Active learning (Settles, 2009). Popular AL methods for classification tasks are pool-based methods using uncertainty sampling (Lewis, 1995) such as entropy (Shannon,

[1]By full control we mean direct control over hyperparameters, architectural extensions, and training regimes.

1948) sampling. Fine-tuning with Active learning can also be enhanced by using adapter modules (Jukić and Šnajder, 2023), Reinforcement Learning (Wertz et al., 2023), or Epistemic Neural networks (ENNs) (Osband et al., 2022). Some methods (especially when fine-tuning with AL methods without any other enhancements) may be less effective when used with LMs due to their inconsistency created by selecting harmful unlearnable outliers or samples that create instability in the optimization process [2] (D'Arcy and Downey, 2022).

One promising idea is to cluster data based on the quality of their representation. For example, compared to the TF-IDF method (Xie et al., 2016), BERT showed better results in many metrics (Subakti et al., 2022). Clustering can also be used for the initialization of AL (Nguyen and Smeulders, 2004) to propagate the classification decision of the classifier trained on representative samples. Clustering can also be performed during the training process after each iteration as in Hassan and Alikhani (2023), where the top ten most informative samples of each cluster are used to train the model.

While prior work has explored architectural modifications, data efficiency techniques, and stability enhancements individually, our work provides a comprehensive empirical investigation into their concurrent application and interaction to enhance low-resource BERT fine-tuning, focusing on the interplay between architectural modifications, data selection efficiency, and fine-tuning stability.

3 Methodologies

3.1 Epistemic neural network

To improve the architecture, we use ENNs that are particularly well-suited for low-resource sentiment analysis because they explicitly model epistemic uncertainty (and differentiate from aleatoric uncertainty), allowing the active learning process to focus on samples where the model is most unsure. This leads to more efficient exploration of the data space and reduces the risk of overfitting to noisy labels. ENNs f_θ consist of two parts: base model (BERT) b_ς and Epinet network e_η. The Epinet, as described in Osband et al. (2023), consists of two Multi-layer perceptrons with one hidden layer each called prior and learnable. The prior network e^P has no trainable parameters and serves to induce some prior knowledge about uncertainty as a variation of the ENN output (Osband et al., 2018). In the learnable network e^L_η, weights are initialized with Glorot initialization (Glorot and Bengio, 2010) and trained to provide meaningful predictions for all probable epistemic index values z. The mentioned process can be described by the equation $f_\theta(x, z) = b_\varsigma(x) + e_\eta(r_\varsigma(x), z)$.

The output of the Epinet e is calculated as the sum of the results of learnable e^L_η and prior network e^P: $e_\eta(r(x), z) = e^P(r(x), z) + e^L_\eta(r(x), z)$.

The Epinet input is comprised of the features of the base network $r_\varsigma(x)$ (in our case the final hidden layer) on an input sample x and an epistemic index z sampled from a standard Gaussian distribution.

3.2 Active learning

We focus on three AL methods: entropy, bald and variance sampling (Osband et al., 2022). These methods called acquisition functions prioritize samples whose labels are most uncertain to the model. For entropy defined as $\mathbb{H}[p] = \sum_x p(x) logp(x)$, the entropy acquisition function is defined as $g^{entropy}(\theta, x) = \mathbb{H}[p(\cdot|x, \theta)]$. The other two methods use an epistemic index to determine uncertainty. Bald acquisition function $g^{bald}(\theta, x) = \mathbb{H}[p(\cdot|\theta, x)] - \int_z P_z(dz)\mathbb{H}[p(\cdot|\theta, x, z)]$ is based on mutual information gain. Variance function $g^{variance}(\theta, x) = \sum_c \int_z P_z(dz)(p(c|\theta, x, z) - p(c|\theta, x))^2$ uses variation in probabilities.

The model is fine-tuned on a task it has not been trained on. Therefore, a new classification random-weighted layer could make the predictions random at first. This phenomenon, known as cold start, can create a situation where the model is assumed to make non-sensical decisions, since it has not yet been trained on any annotated data (Jin et al., 2022). To examine this, we fine-tune the model in both cold (classic fine-tuning) and "warm" manner (first epoch uses half of the dataset at random).

3.3 Data sampling

Data sampling in AL can be performed after each training epoch or after each training step (Settles, 2009), affecting both the frequency of data acquisition and the computational cost (e.g., loading a subset vs the entire dataset). In experiments, we adopt training-step sampling. Departing from the conventional AL framework, we introduce two distinct fine-tuning approaches: `Accumulating` and `Recalculating`. In the `Accumulating` approach,

[2] The definition of instability is taken from D'Arcy and Downey (2022) to be the variance in test set accuracy observed when training multiple different random-seeded models on the same set of data.

all data sampled in previous epochs are retained and used for subsequent fine-tuning aligning with standard AL practices. The `Recalculating` approach re-samples data in each epoch independently, as if each were the initial training iteration. This design gives the model greater flexibility to select informative data at each stage. While `Accumulating` fine-tuning is widely used, `Recalculating` fine-tuning represents a novel contribution. Further details on these methods are provided in Appendix A.

3.4 Clustering

To support the findings of Hu et al. (2010), we apply Agglomerative Hierarchical Clustering (Voorhees, 1986) using Ward's linkage, shown effective in capturing semantic relationships in text embeddings (Sharma et al., 2019). Clustering starts with each sample's embedding as a separate cluster, sequentially merging them bottom-up. We applied clustering in two modes: (1) init clustering, performed before fine-tuning to sample from each cluster, and (2) dynamic clustering, repeated before each epoch similarly to Hassan and Alikhani (2023).

To test the dependence of acquisition functions on the model itself, we also propose `Furthest-batch` acquisition function that selects samples furthest from the medoid (sample closest to centroid) of cluster. This function is motivated by the hypothesis that samples located far from the medoid represent boundary cases or diverse viewpoints within the cluster that are not necessarily outliers. By prioritizing these samples, we aim to improve the model's ability to generalize to unseen data. It also enables comparison with uncertainty-based methods and may reveal the extent to which fine-tuning depends on the model vs. dataset.

3.5 Active learning scheduling

Similarly to Gonsior et al. (2024) but more generalizing, we introduce several Active Learning schedulers that modulate the number of samples used. We hypothesize that this may reduce reliance on harmful outliers, minimize annotation costs, and help prevent issues like performance degradation.

The samples are passed through the model to obtain feature vectors, which are evaluated by the AL method to yield values (`AL results`) and the corresponding indices indicating sample ranking (`AL order`). The base scheduler selects 75% of the data according to `AL order`, functioning similarly to standard AL. The prob scheduler also selects 75%, but samples without replacement using normalized `AL results` as probabilities.

Linear schedulers reduce annotation usage progressively after each epoch. They initially select 50% of the data for both warm and cold fine-tuning, decreasing to 20%, 15%, 10%, and finally 5%. Like the base scheduler, they either select deterministically by `AL order` (linear scheduler) or probabilistically (linear prob scheduler).

Dif-build and dif-build-unique schedulers operate differently. They sort `AL results`, compute the average pairwise difference, and locate the first index where the next difference exceeds this average. This index sets the sampling threshold. Sampling proceeds as in prob schedulers, either with or without replacement, forming different variants.

4 Experiments

We trained the models for 5 epochs for the sentiment analysis task while using the Adam optimizer (Kingma, 2014). All models are based on the BERT architecture and pre-trained in the respective lower-resourced language, namely SlovakBERT (Pikuliak et al., 2021), IceBERT (Snæbjarnarson et al., 2022), BERTu (Micallef et al., 2022) and BERTurk models (Schweter, 2020). All hyperparameters are detailed in the Appendix A. As baselines, we employ the classic BERT fine-tuning approach, wherein a pre-trained BERT model is further trained on the downstream task with task-specific labels.

The assumption of low-resource setting holds true without modifications in experiments with languages spoken by small amounts of people, such as Maltese. To simulate a low-resource scenario, we intentionally down-sample the dataset for languages such as Turkish, as described in Appendix A. This enables controlled comparison across languages and aligns with recent approaches.

Language	Size	Split	#Classes
Slovak	30k	83-8.5-8.5	3
Maltese	851	70-10-20	2
Icelandic	25k	80-10-10	2
Turkish	25k	80-10-10	3

Table 1: Specifications of datasets and their splits.

For evaluation, we used monolingual sentiment analysis datasets detailed in Table 1. In experiments, we fine-tune each model three times and report the average. More details are presented in the Appendix A.

5 Results

The key results of our study are summarized below, with additional details provided in Appendix A. Table 2 presents a comparison between the base model and the best-performing AL model based on F1 score. Models using Active Learning largely outperform baseline models. We also performed a Student's paired t-test to evaluate the statistical significance of our findings, with a significance level of $p < 0.05$. All top-performing AL models incorporate both clustering and scheduling strategies.

Model	Baseline F1	Best AL F1
SlovakBERT	70.98	71.83
BERTu	82.08	86.10†
IceBERT	77.89	80.06†
BERTurk	91.06	92.16†

Table 2: Results of the best AL models according to the F1 score compared to their respective baselines, with † indicating statistical significance.

Figure 1 shows the dataset fraction needed to reach the F1 scores in Table 2, suggesting that data usage often drops markedly even as performance improves. Even more, to reach at least the baseline performance with Active Learning models, amount of data samples needed reduce drastically. We present this fact with Figure 2. It is apparent that if the fine-tuning is more focused on saving data annotations rather than enhancing performance (but still keeping it at least comparable), the AL is far superior to the classic fine-tuning process. For more details see Appendix A.4.

Experiments also indicate that AL contributes to more stable fine-tuning, with fewer instances of performance degradation. This is presented in Figure3, where for the sake of clarity we show the comparison of SlovakBERT base model and best AL model performance comparison.

Our results show that the proposed methods are most effective when combined. To assess individual component contributions in Active Learning, we conducted controlled experiments with the BERTurk model, keeping all settings constant (accumulating sampling, cold start, vanilla BERT architecture, no clustering, no scheduling, and entropy acquisition) while modifying one component at a time. Table 3 presents the best performing configurations based on the F1 score.

Finally, we also present a comparison of the acquisitions functions. In Figure 4, we show a comparison of the best IceBERT models using different acquisition functions. Results indicate that using any acquisition function apart from Furthest-batch is beneficial to the fine-tuning while differences among them are very small. Furthest-batch seems to have similar performance as the baseline model but with much less data used (see Appendix A.4 for more information). Interestingly, while fine-tuning for more epochs seems to be crucial for most of the acquistion function to enhance the performance to the maximum, baseline model started to slightly decline after just 2 epochs.

To evaluate parameter-efficient strategies, we conducted controlled experiments using LoRA adapters (Hu et al., 2022), keeping data splits and training sizes identical to the full fine-tuning baseline. We explored multiple configurations by varying LoRA rank, learning rate, and dropout. Across all languages and setups, LoRA consistently underperformed full fine-tuning, and performance de-

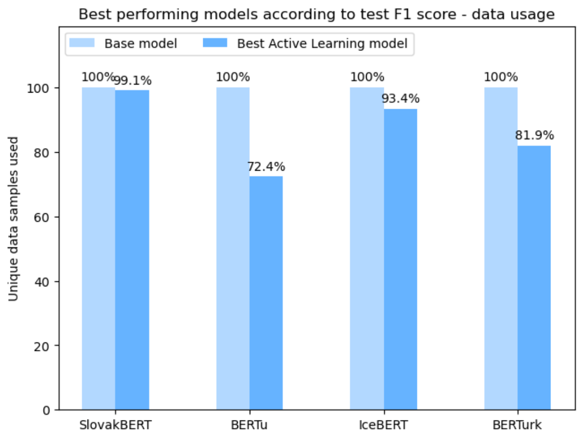

Figure 1: Data used in the fine-tuning of the models.

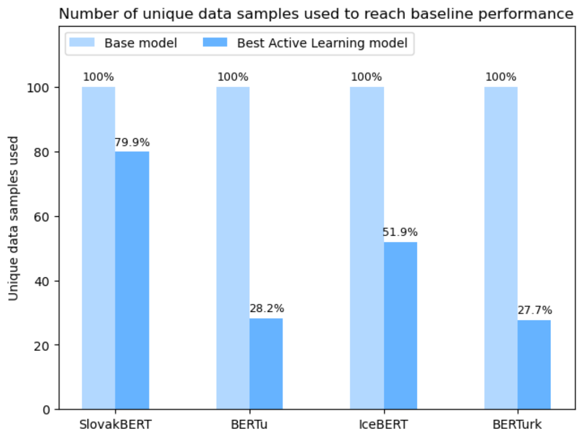

Figure 2: Data used in the fine-tuning of the models reaching baseline performance while using fewest data samples possible.

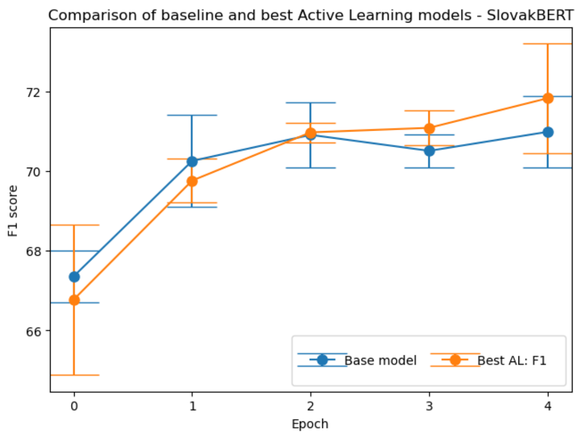

Figure 3: Learning curves of SlovakBERT models.

Characteristic	F1 score	% of data
Dynamic clustering	91.85	96.81%
Recalculating sampling	91.70	81.86%
Prob scheduler	91.31	97.54%
Bald acquisition	91.15	96.4%
ENN architecture	91.14	97.54%
Linear scheduler	91.10	61.49%

Table 3: Comparison of the best BERTurk results of different fine-tuning settings.

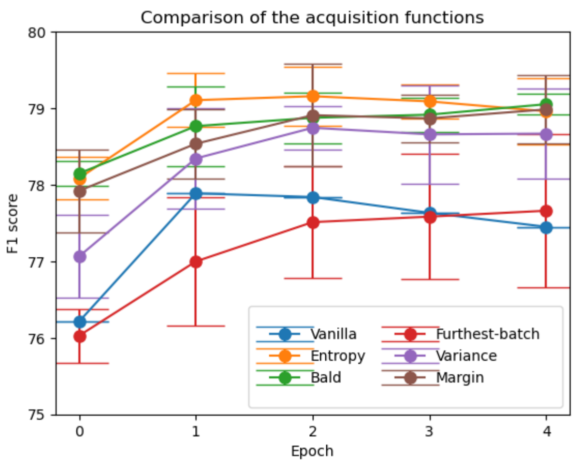

Figure 4: Learning curves of IceBERT models with different acquistion functions.

Language	Best AL	Best LoRA
Slovak	+1.20%	-5.11%
Maltese	+4.90%	-51.00%
Icelandic	+2.79%	-0.60%
Turkish	+1.21%	-0.96%

Table 4: Relative performance changes in F1 score using Active Learning or LoRA when compared to the respective baseline models.

graded substantially on the low-resource Maltese condition. These results contrast with prior work reporting strong LoRA performance on large-scale English benchmarks, indicating that the parameter-efficient updates were not expressive enough to model the language-specific sentiment patterns required in our experiments.

6 Conclusions

Our findings demonstrate that integrating Active Learning with clustering and scheduling yields significant improvements for low-resource language models. This systematic approach reduced annotation requirements by up to 27.6% for IceBERT and 18.1% for BERTurk while simultaneously improving performance, with statistically significant gains in the F1 score of up to 4.02 points for BERTu and 2.17 points for IceBERT. Furthermore, combined methodology notably improved fine-tuning stability, reducing performance fluctuations across training epochs. While effectiveness varies by language and dataset characteristics, empirical evidence consistently supports the value of our approach, possibly generalizing to medium and high-resource languages such as Turkish, where particularly the recalculating sampling strategy and linear scheduler achieved a strong result with 81.9% and 61.49% of training data, respectively. Moreover, the proposed fine-tuning pipeline can be easily extended to other classification tasks, such as Named Entity Recognition, where only slight changes to architecture and acquisition functions are needed.

7 Limitations

7.1 Datasets

The datasets used in this work were expected to be reasonably small, either by choice (e.g., using a portion of a large dataset) or by necessity. It is understandable that the size of the dataset influences expected savings on annotations. In this work, 2 of the datasets can already be considered small. From two of the datasets with a reasonable amount of data, only a smart portion was used to show the possible enhancements presented in this work.

7.2 Models

In comparison to state-of-the-art architectures, the BERT and similar models used in our experiments are only a fraction of their size. Nevertheless, for fine-tuning, low-resource languages do not have many other choices. As BERT models are still relatively popular, the decision to use them in our work was made, although we acknowledge that for some (or most) use cases, using bigger decoder-style models can obtain better performance.

7.3 Experiments

For each combination of model's important fine-tuning characteristic (AL method, AL scheduling, clustering, ...), we conducted three runs and calculated the average that represents the final result. While this aims to eliminate model bias, a larger number of runs might help reduce it much more.

For Turkish and Icelandic, we simulate low-resource conditions via controlled downsampling, which, while enabling fair comparison, may not fully capture the challenges of genuine low-resource environments.

Due to computational constraints, we were unable to explore all possible combinations of AL methods, clustering techniques, and scheduling strategies. Future work should investigate a wider range of configurations, particularly those excluded from this study as mentioned in Appendix A.3.

7.4 Results

We argue that the presented results show that using Active Learning with clustering and scheduling enhances the fine-tuning process of lower-resourced language BERT models while also enhancing F1 scores. Further experiments on more languages could strenghten this claim; nevertheless, our results on bigger and more used language (Turkish) can serve as a form of generalization on also medium-resource languages.

The differences observed across experimental setups are likely attributable to the specific characteristics of each language model and dataset. While our results demonstrate that Active Learning and clustering generally lead to improved performance, we acknowledge that tailoring acquisition functions and scheduling strategies to individual model–dataset combinations can yield even greater gains. Therefore, it is highly plausible that conducting targeted experiments with varied acquisition functions and scheduling approaches is always beneficial for identifying the most effective configuration in each specific context.

Acknowledgements

This work was supported by grant APVV-21-0114.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Ahmad Al-Sallab, Ramy Baly, Hazem Hajj, Khaled Bashir Shaban, Wassim El-Hajj, and Gilbert Badaro. 2017. Aroma: A recursive deep learning model for opinion mining in arabic as a low resource language. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, 16(4):1–20.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the association for computational linguistics*, 5:135–146.

Keith Cortis and Brian Davis. 2019. A social opinion gold standard for the Malta government budget 2018. In *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, pages 364–369, Hong Kong, China. Association for Computational Linguistics.

Mike D'Arcy and Doug Downey. 2022. Limitations of active learning with deep transformer language models.

Jacob Devlin. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Alexiei Dingli and Nicole Sant. 2016. Sentiment analysis on maltese using machine learning. In *Proceedings of The Tenth International Conference on Advances in Semantic Processing (SEMAPRO 2016)*, pages 21–25.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings.

Julius Gonsior, Christian Falkenberg, Silvio Magino, Anja Reusch, Claudio Hartmann, Maik Thiele, and Wolfgang Lehner. 2024. Comparing and improving active learning uncertainty measures for transformer models by discarding outliers. *Information systems frontiers*, pages 1–17.

Sabit Hassan and Malihe Alikhani. 2023. D-calm: A dynamic clustering-based active learning approach for mitigating bias. *arXiv preprint arXiv:2305.17013*.

Benjamin Heinzerling and Michael Strube. 2017. Bpemb: Tokenization-free pre-trained subword embeddings in 275 languages. *arXiv preprint arXiv:1710.02187*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. 2022. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3.

Rong Hu, Brian Mac Namee, and Sarah Jane Delany. 2010. Off to a good start: Using clustering to select the initial training set in active learning. In *Twenty-Third International FLAIRS Conference*.

Qiuye Jin, Mingzhi Yuan, Shiman Li, Haoran Wang, Manning Wang, and Zhijian Song. 2022. Cold-start active learning for image classification. *Information sciences*, 616:16–36.

Josip Jukić and Jan Šnajder. 2023. Parameter-efficient language model tuning with active learning in low-resource settings. *arXiv preprint arXiv:2305.14576*.

Diederik P Kingma. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

David D Lewis. 1995. A sequential algorithm for training text classifiers: Corrigendum and additional data. In *Acm Sigir Forum*, volume 29, pages 13–19. ACM New York, NY, USA.

Kurt Micallef, Albert Gatt, Marc Tanti, Lonneke van der Plas, and Claudia Borg. 2022. Pre-training data quality and quantity for a low-resource language: New corpus and BERT models for Maltese. In *Proceedings of the Third Workshop on Deep Learning for Low-Resource Natural Language Processing*, pages 90–101, Hybrid. Association for Computational Linguistics.

Hieu T Nguyen and Arnold Smeulders. 2004. Active learning using pre-clustering. In *Proceedings of the twenty-first international conference on Machine learning*, page 79.

Hellina Hailu Nigatu, Atnafu Lambebo Tonja, Benjamin Rosman, Thamar Solorio, and Monojit Choudhury. 2024. The zeno's paradox oflow-resource'languages. *arXiv preprint arXiv:2410.20817*.

Ian Osband, Seyed Mohammad Asghari, Benjamin Van Roy, Nat McAleese, John Aslanides, and Geoffrey Irving. 2022. Fine-tuning language models via epistemic neural networks. *arXiv preprint arXiv:2211.01568*.

Ian Osband, John Aslanides, and Albin Cassirer. 2018. Randomized prior functions for deep reinforcement learning. *Advances in Neural Information Processing Systems*, 31.

Ian Osband, Zheng Wen, Seyed Mohammad Asghari, Vikranth Dwaracherla, Morteza Ibrahimi, Xiuyuan Lu, and Benjamin Van Roy. 2023. Epistemic neural networks. *Advances in Neural Information Processing Systems*, 36:2795–2823.

Matúš Pikuliak, Štefan Grivalský, Martin Konôpka, Miroslav Blšták, Martin Tamajka, Viktor Bachratý, Marián Šimko, Pavol Balážik, Michal Trnka, and Filip Uhlárik. 2021. Slovakbert: Slovak masked language model. *arXiv preprint arXiv:2109.15254*.

Stefan Schweter. 2020. Berturk - bert models for turkish.

Burr Settles. 2009. Active learning literature survey.

Claude Elwood Shannon. 1948. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423.

Shweta Sharma, Neha Batra, et al. 2019. Comparative study of single linkage, complete linkage, and ward method of agglomerative clustering. In *2019 international conference on machine learning, big data, cloud and parallel computing (COMITCon)*, pages 568–573. IEEE.

Jiho Shin, Clark Tang, Tahmineh Mohati, Maleknaz Nayebi, Song Wang, and Hadi Hemmati. 2023. Prompt engineering or fine tuning: An empirical assessment of large language models in automated software engineering tasks. *arXiv preprint arXiv:2310.10508*.

David Short. 2018. Czech and slovak. In *The world's major languages*, pages 314–338. Routledge.

Vésteinn Snæbjarnarson, Haukur Barri Símonarson, Pétur Orri Ragnarsson, Svanhvít Lilja Ingólfsdóttir, Haukur Páll Jónsson, Vilhjálmur Þorsteinsson, and Hafsteinn Einarsson. 2022. A warm start and a clean crawled corpus–a recipe for good language models. *arXiv preprint arXiv:2201.05601*.

Michal Štefánik, Marek Kadlčík, Piotr Gramacki, and Petr Sojka. 2023. Resources and few-shot learners for in-context learning in slavic languages. *arXiv preprint arXiv:2304.01922*.

Alvin Subakti, Hendri Murfi, and Nora Hariadi. 2022. The performance of bert as data representation of text clustering. *Journal of big Data*, 9(1):15.

Ellen Marie Voorhees. 1986. *The effectiveness and efficiency of agglomerative hierarchic clustering in document retrieval*. Cornell University.

Lukas Wertz, Jasmina Bogojeska, Katsiaryna Mirylenka, and Jonas Kuhn. 2023. Reinforced active learning for low-resource, domain-specific, multi-label text classification. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 10959–10977.

Junyuan Xie, Ross Girshick, and Ali Farhadi. 2016. Unsupervised deep embedding for clustering analysis. In *International conference on machine learning*, pages 478–487. PMLR.

A Appendix

A.1 Datasets

The largest dataset used was the Slovak ČSFD dataset (Štefánik et al., 2023) with 30k annotated samples containing movie reviews. Although the original dataset includes six sentiment labels, we simplify the classification task to three classes by applying the following mapping: [0, 1 $\rightarrow$ 0], [2, 3 $\rightarrow$ 1], and [4, 5 $\rightarrow$ 2].

While Turkish is not a low-resource language, we intentionally downsampled the dataset comprising tweets and other sentiment-related text samples to 5% to investigate the performance of active learning methods in extremely data-scarce scenarios. This allowed us to evaluate the robustness of our approach when annotation budgets are severely limited.

For the Maltese language, we use the joint datasets of (Cortis and Davis, 2019) and (Dingli and Sant, 2016) containing data originating from comments on news articles and social media posts. Similarly as in Turkish dataset, in Icelandic dataset, which was created by translating IMDB movie reviews, we down-sample to 25k annotated data samples to mimic low-resource scenario [3]. Selection of the relatively small and specialized datasets stems from the setup of our research task, which tries to mimic many real-life scenarios (particularly in the context of lower-resourced languages), in which only a small task-specific dataset is available and/or where expert annotation is very costly. This in turn creates the need to make maximal use of the available data. Additional details about datasets are presented in Table 5.

Dataset	**Class 1**	**Class 2**	**Class 3**	**Avg token number**
Slovak	33.2%	33.4%	33.4%	368
Maltese	60%	40%	-	123
Icelandic	50%	50%	-	1367
Turkish	53.5%	34.9%	11.6%	140

Table 5: Specifications of datasets' class distribution and average token length.

A.2 Models

Some of the most important specifications of the models used can be seen in Table 6:

Characteristic\Model	SlovakBERT	IceBERT	BERTurk	BERTu
Architecture	RoBERTa	RoBERTa	BERT	BERT
#Layers	12	12	12	12
#Heads	12	12	12	12
Hidden size	768	768	768	768
#Parameters	125M	125M	109M	560M
#Training tokens	4.6B	2.7B	44.04B	500M
#Training steps	300K	225K	2M	1M
Vocabulary size	50K	50K	32K	52K
Pretraining data size	19.35GB	15.8GB	35GB	2.52GB

Table 6: Important specifications of models used in experiments.

There are two main reasons why we use BERT-based models:

- For many lower-resourced languages, BERT-based models are the best (and often the only) available Pre-trained Language Model option.
- The model family is fairly easy to use, as well as resource-efficient, and as such has a higher chance to reach a wider research audience.

[3]All datasets are available on Huggingface website, namely Slovak sk_csfd-movie-reviews, Turkish turkish-sentiment-analysis-dataset, Maltese maltese_sa and and Icelandic imdb-isl-mideind-translate.

A.3 Experiments

Table 7 presents important parameters for our low-setting environment:

Parameter	**Value**
Epochs	5
Max text length	64
Batch size	32
Learning rate	$5 * 10^{-5}$

Table 7: Specifications of parameters used in experiments.

If we assume that at the start of the training (in our case fine-tuning) we have an unlabeled pool U_0 with all data samples from the dataset and an empty annotated data pool D_0, the difference in one training epoch between Accumulating and Recalculating fine-tuning can be summarized as:

Accumulating (building) fine-tuning

1. Load data samples from previous epoch unlabeled data pool U_{e-1} and used annotated data pool D_{e-1}.

2. Sample up to 75% new data samples from U_{e-1}, annotate them and add them to D_{e-1}, thus creating new data pools U_e (without newly sampled data) and D_e (with newly sampled data).

3. Fine-tune the model with all the accumulated data in the dataset D_e.

Recalculating fine-tuning

1. Sample up to 75% of the data samples from U_0, annotate them and add them to empty D_0, while retaining the same data pool U_0 and creating the data pool D_e (with sampled data).

2. Fine-tune the model with all the data in the dataset D_e.

3. Delete the whole D_e.

As the model's weights are changing and the model is learning, it can decide that some of the data samples used previously can be no longer helpful or even misleading (e.g. outliers). This fact is not relevant in Accumulating fine-tuning, which uses these samples nevertheless, but serves as a motivation for Recalculating fine-tuning. One important fact of the Recalculating fine-tuning: since this happens in multiple epochs, some data can be chosen multiple times (and some will not be chosen at all); still, most of the dataset or the whole dataset may be used (more precisely: the data sample can be chosen multiple times, but it will be used at most once during each training epoch since the annotated dataset is built from fresh every epoch.).

Some of the combinations of acquisition function, architecture, clustering, and scheduling are not possible or helpful due to different circumstances. To save on computational resources, their exploration was not conducted. In Tables 8, 9 and 10, we show which of the combinations were not present in our experimental setups:

Option	No clustering	Init clustering	Dynamic clustering
Base / prob scheduler	✓	✓	✓
Linear schedulers	✓	✥	✓
Dif-build schedulers	✗	✓	✓

Table 8: Setups of different clusterings and schedulers used in the experiments.

Due to the design of dif-build schedulers, it is appropriate to evaluate their performance only on data drawn from the same clusters, rather than from random samples across the entire dataset. Init

Option	Classic AF	Epistemic AF	Furthest-batch AF
Accumulating base BERT	✓	✗	✓
Accumulating ENN	✓	✓	✓
Recalculating base BERT	✓	✗	✓
Recalculating ENN	✓	✓	✓

Table 9: Setups of different acquisition functions and architecture types used in the experiments.

Option	Classic AF	Epistemic AF	Furthest-batch AF
Base scheduler	✓	✓	✓
Prob scheduler	✓	✓	✗
Linear scheduler	✓	✓	✓
Linear prob scheduler	✓	✓	✗
Dif-build schedulers	✓	✓	✗

Table 10: Setups of different acquisition functions and schedulers used in the experiments.

clustering plays a role solely during the first epoch of fine-tuning. Consequently, combining it with linear schedulers—which adjust the volume of sampled data across epochs—offers no advantage over dynamic clustering (marked as ✣).

Epistemic acquisition functions inherently rely on the epistemic index and the broader Epistemic neural network architecture. In contrast, the furthest-batch acquisition function is based exclusively on data sample characteristics. As a result, model-dependent schedulers such as prob, linear prob and dif-build are not applicable when using furthest-batch. Additionally, because furthest-batch determines acquisition based on distances from medoids, it requires clustering and cannot operate without it.

Although combining furthest-batch with dynamic clustering is feasible, it does not offer any benefit over using it with init clustering. This is because the selected samples remain unchanged across epochs, rendering dynamic updates unnecessary in this case.

A.4 Results

This section presents supplementary results not included in the main text. In Section 5, we compared the best-performing models against their respective baselines and analyzed the individual contributions of key components of Active Learning (AL). Here, we extend this analysis by evaluating how different aspects of AL, when combined, influence fine-tuning performance—reflecting more realistic usage scenarios.

Specifically, we report the average F1 score gains or losses of models incorporating particular AL characteristics (in combination with others), relative to their corresponding baselines. These comparisons are aggregated across all models and languages, along with the percentage of labeled data utilized. After that, we report individual statistics as in Table 3 for other languages (Slovak, Icelandic, Maltese).

Subsequently, we present the learning curves for BERTu, IceBERT, and BERTurk models to illustrate performance trends throughout the fine-tuning process. Finally, to facilitate direct comparisons of various fine-tuning configurations, we include a series of tables showing the best results per language for the following model variants: (i) the vanilla base model, (ii) the model with only Active Learning, (iii) Active Learning with clustering, and (iv) Active Learning with both clustering and scheduling.

In Table 11 we present the results of an ablation study designed to compare the impact of various fine-tuning strategies, including different schedulers, clustering methods, and other design choices. Notably, the results highlight the significant role of Active Learning. All tested acquisition functions outperformed the baseline condition without AL, underscoring its overall effectiveness. Among them, the Bald acquisition function achieved the highest F1 scores, while the furthest-batch strategy led to substantial reductions in the number of required annotations.

Fine-tuning BERT models as Epistemic neural networks contributed to both improved performance and reduced data requirements. Similarly, cold fine-tuning yielded beneficial effects, challenging the assumption that possible random fine-tuning during the initial epoch degrades performance. Additionally,

Characteristic	**+- % of F1 score**	**% of data**
	Architecture	
Vanilla	+1.89%	90.72%
ENN	+2.39%	87.59%
	Start	
Warm	+1.74%	93.06%
Cold	+2.44%	89.12%
	Sampling	
Recalculating	+2.26%	86.43%
Accumulating	+2.39%	92.54%
	Acquisition function	
None	-2.67%	99.48%
Entropy	+2.28%	92.07%
Bald	+2.32%	87.89%
Variance	+2.26%	89.72%
Furthest-batch	+1.68%	79.81%
	Clustering	
No clustering	+1.76%	95.51%
Init clustering	+2.26%	78.66%
Dynamic clustering	+2.21%	97.12%
	Schedulers	
Base	+2.02%	85.67%
Prob scheduler	+2.24%	92.76%
Linear scheduler	+2.08%	80.68%
Linear prob scheduler	+1.66%	81.68%
Dif-build	+1.59%	90.25%
Dif-build-unique	+1.95%	90.10%

Table 11: Comparison of the average results of different fine-tuning settings.

recalculating sampling showed comparable performance to traditional accumulation-based sampling, while offering further annotation savings.

Finally, the use of linear schedulers—particularly the standard linear scheduler—proved effective in balancing annotation efficiency with strong model performance.

Characteristic	**F1 score**	**% of data**
ENN architecture	71.43	99.52%
Dynamic clustering	71.24	99.62%
Recalculating sampling	71.17	94.2%

Table 12: Comparison of the best SlovakBERT results of different fine-tuning settings.

Characteristic	**F1 score**	**% of data**
Dynamic clustering	85.38	98.9%
Init clustering	84.52	77.98%
Entropy acquisition	84.35	96.02%

Table 13: Comparison of the best BERTu results of different fine-tuning settings.

Characteristic	**F1 score**	**% of data**
Bald acquisition	79.77	93.37%
ENN architecture	79.22	93.37%
Prob scheduler	78.23	99.5%

Table 14: Comparison of the best ICEBert results of different fine-tuning settings.

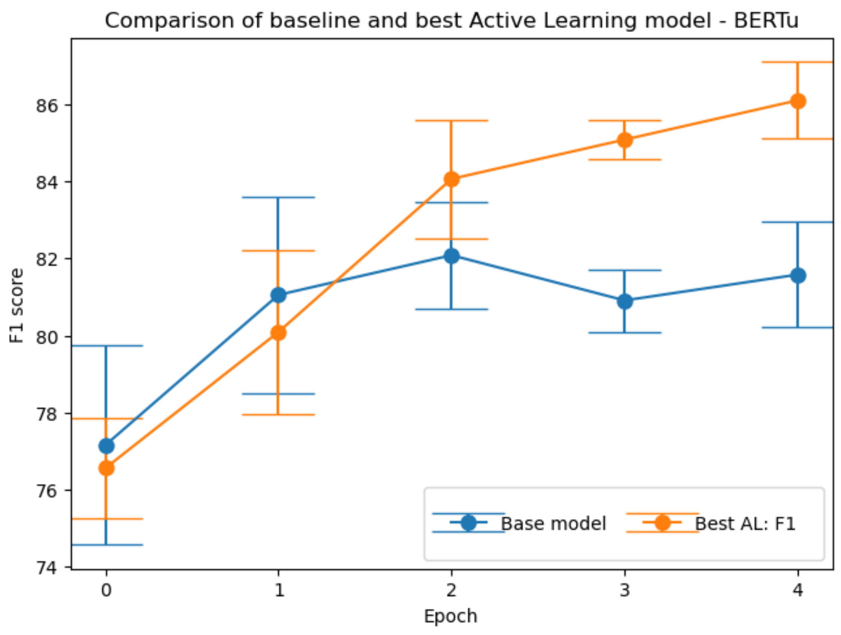

Figure 5: Learning curves of baseline model and the best performing AL models while fine-tuning BERTu.

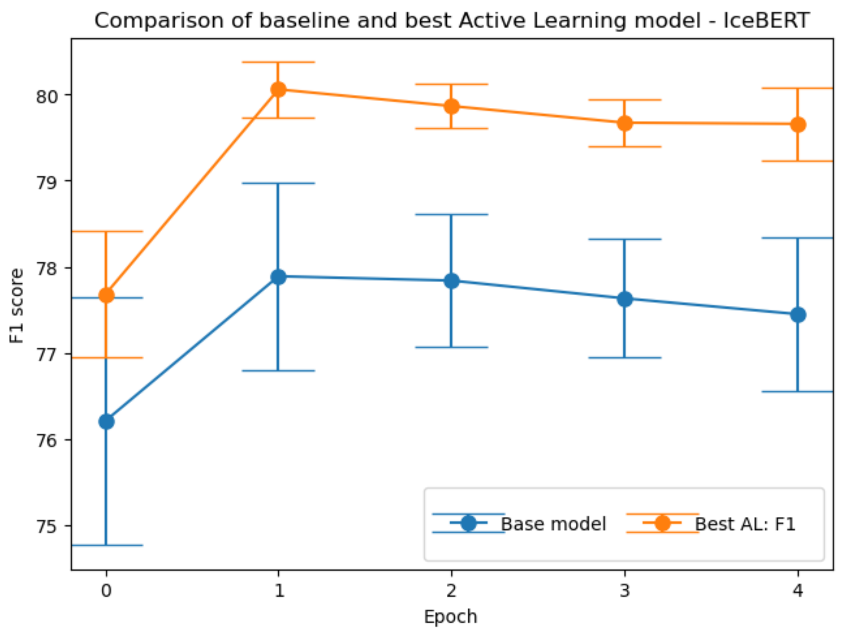

Figure 6: Learning curves of baseline model and the best performing AL models while fine-tuning IceBERT.

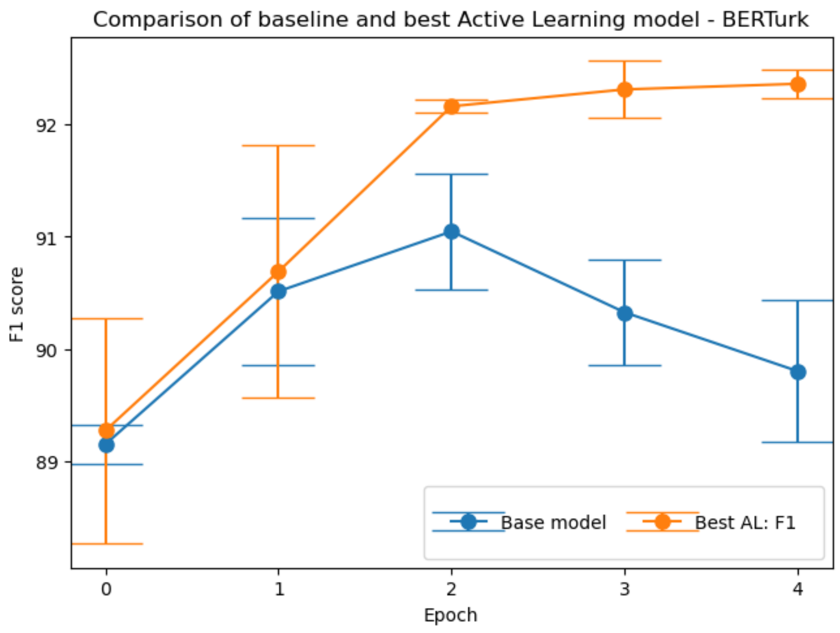

Figure 7: Learning curves of baseline model and the best performing AL models while fine-tuning BERTurk.

Model	**F1 score**	**% of data**
Vanilla	71.17	100%
Entropy	71.23	99.92%
Bald + dynamic	71.64	99.6%
Entropy + dynamic + linear prob	71.83	99.1%

Table 15: Comparison of the best SlovakBERT models.

Model	**F1 score**	**% of data**
Vanilla	82.76	100%
Entropy	84.35	96.7%
Bald + base	85.5	78%
Bald + init + linear	86.1	72.4%

Table 16: Comparison of the best BERTu models.

Model	**F1 score**	**% of data**
Vanilla	78.07	100%
Bald	79.77	93.4%
Bald + init	78.89	85.3%
Entropy + dynamic + prob	80.06	93.4%

Table 17: Comparison of the best IceBERT models.

Model	**F1 score**	**% of data**
Vanilla	91.06	100%
Entropy	91.3	81.9%
Bald + init	91.94	70.13%
Bald + dynamic + prob	92.16	81.9%

Table 18: Comparison of the best BERTurk models.