Your Pretrained Model Tells the Difficulty Itself: A Self-Adaptive Curriculum Learning Paradigm for Natural Language Understanding

Qi Feng[1]* **Yihong Liu**[1,2]* **Hinrich Schütze**[1,2]
[1]Center for Information and Language Processing, LMU Munich
[2]Munich Center for Machine Learning (MCML)
fengqi928@outlook.com

Abstract

Curriculum learning is a widely adopted training strategy in natural language processing (NLP), where models are exposed to examples organized by increasing *difficulty* to enhance learning efficiency and performance. However, most existing approaches rely on manually defined difficulty metrics – such as text length – which may not accurately reflect the model's own perspective. To overcome this limitation, we present a self-adaptive curriculum learning paradigm that prioritizes fine-tuning examples based on difficulty scores predicted by pre-trained language models (PLMs) themselves. Building on these scores, we explore various training strategies that differ in the ordering of examples for the fine-tuning: from easy-to-hard, hard-to-easy, to mixed sampling. We evaluate our method on four natural language understanding (NLU) datasets covering both binary and multi-class classification tasks. Experimental results show that our approach leads to faster convergence and improved performance compared to standard random sampling. We make our code publicly available.[1]

1 Introduction

Although large language models (LLMs) are highly valued in the NLP community for their broad capabilities (Naveed et al., 2024; Chang et al., 2024), their substantial computational cost often makes them impractical for many real-world scenarios – particularly for simple classification tasks that require rapid responses or deployment on resource-constrained infrastructure (Bai et al., 2024; Cunningham et al., 2024). As a result, *task-specific* NLP models – those pre-trained and subsequently fine-tuned on labeled data for specific tasks, e.g., sentiment analysis – remain highly relevant (Zhao et al., 2024b). While many studies have focused on enhancing the effectiveness of pre-training (Du et al., 2021; Yu et al., 2022a; Liu et al., 2024; Hu et al., 2024), the high resource demands of this stage make it more practical to instead develop improved fine-tuning strategies (Xu et al., 2020; Chen et al., 2021; Hu et al., 2022a; Ding et al., 2023).

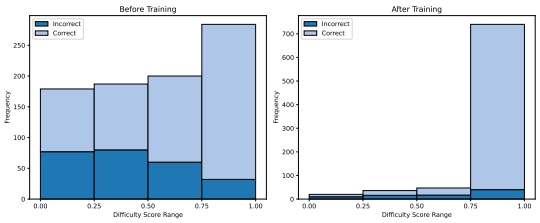

Figure 1: Frequencies of samples being incorrectly (dark blue) and correctly classified (light blue) by BERT before and after 1 epoch of training. The model tends to make worse decisions when samples are difficult and better decisions when they are easy. Note that a sample with a difficulty score of 0 is the most difficult one.

One important class of fine-tuning strategies centers around the concept of *curriculum* – a process inspired by human learning. *Curriculum Learning*, first introduced by Bengio et al. (2009) in the general machine learning domain, has since demonstrated effectiveness in NLP tasks as well (Xu et al., 2020; Zhu et al., 2021; Maharana and Bansal, 2022; Ranaldi et al., 2023; Gao et al., 2024). This paradigm involves structuring training data from simpler to more complex examples, enabling models to build knowledge incrementally and learn more efficiently. A central challenge in applying curriculum learning lies in defining *difficulty*. Most prior work estimates difficulty using surface-level features such as sentence length or word rarity (Platanios et al., 2019; Xu et al., 2020; Ranaldi et al., 2023). However, these metrics may not align with the model's internal understanding – especially for PLMs capable of capturing deeper semantic attributes like irony or ambiguity thanks to massive pre-training. Moreover, the assumption that training should always progress from easy to hard is

*Equal contribution.
[1]https://github.com/alitanokiki/self-adaptive-curriculum-nlu-acl2025

debatable; models may benefit from early exposure to difficult examples or from revisiting easier ones in training to mitigate forgetting (Kirkpatrick et al., 2017; Ke et al., 2021; Huang et al., 2024).

To this end, we propose a self-adaptive curriculum learning paradigm that explores various sampling strategies driven by the model's own confidence. Rather than relying on manually defined difficulty heuristics based on the surface feature of an example, we leverage the PLM itself to compute a difficulty score – specifically, a confidence measure that reflects how certain the model is when classifying an example using a prompt template and a verbalizer component (Schick and Schütze, 2021a). For each example, we define its difficulty as the maximum absolute difference among the predicted class probabilities, where a smaller difference indicates greater uncertainty (i.e., higher difficulty). Since this computation requires no parameter updates, it can be performed efficiently across the dataset. Once difficulty scores are computed, we sort the examples in ascending or descending order and explore three categories of sampling strategies: ***Naive sequential sampling***: examples are selected in order from easiest to hardest, or in reverse. ***Probability-based sampling***: examples are sampled probabilistically, with sampling probabilities defined based on their difficulty ranks. ***Partitioned batch sampling***: examples are divided into easy and hard groups, and batches are formed by sampling from both partitions during fine-tuning.

To validate our proposed methodology, we conduct extensive experiments on four NLU datasets covering both binary and multi-class classification tasks, including sentiment analysis, hate speech detection, and natural language inference. We show that the difficulty scores predicted by the PLM itself serve as a reliable proxy for model uncertainty – examples with higher difficulty scores are much more likely to be misclassified, as shown in Figure 1. Moreover, our sampling strategies yield competitive or superior performance compared to standard random sampling in the full-dataset fine-tuning setting. In the few-shot fine-tuning setting, our methods generally outperform the baseline methods, demonstrating strong generalization and robustness. Our contributions are as follows:

(i) We propose a self-adaptive curriculum paradigm that prioritizes fine-tuning examples based on difficulty scores predicted by the PLM itself. (ii) We propose three categories of sampling strategies based on ranked lists of examples according to their difficulty scores. (iii) We empirically validate our approach on four diverse NLU tasks, achieving strong results in both full-dataset and few-shot fine-tuning scenarios.

2 Related Work

2.1 Sampling Strategies

Traditional random sampling methods, though widely used, often fail to make the model learning more effective. Therefore, more advanced sampling strategies have been explored, including strategies with stratified sampling (Neyman, 1934; Qian et al., 2009), multistage sampling (Nadeem et al., 2020), adaptive ranking-based sampling (Song et al., 2022) and class balancing techniques such as balanced data sampling (Shao et al., 2024). Active learning (AL) selects the most informative instances for annotation (Lewis and Gale, 1994) to better leverage unlabeled data, with recent strategies including uncertainty-based sampling (Yu et al., 2022b), cold-start AL via masked language modeling loss (Yuan et al., 2020), self-active learning for multilingual settings (Dossou et al., 2022), and hybrid AL combining uncertainty and diversity (Azeemi et al., 2025). A comprehensive survey of AL in NLP is provided by Zhang et al. (2022). Adaptive sampling techniques, which dynamically adjust sample selection during training, recent research includes difficulty-aware negative sampling (Li et al., 2019), hard negative mining in extreme classification (Dahiya et al., 2023), and class-adaptive re-sampling to mitigate false negatives in weak supervision (Tan et al., 2023).

2.2 Curriculum Learning

Curriculum learning (CL) (Bengio et al., 2009) defines the difficulty of the sample and improves model convergence and performance by ordering training samples from easy to hard (Soviany et al., 2022). In NLP, it can be implemented by sorting and sampling sentences based on features such as sentence length or word rarity (Platanios et al., 2019). However, empirical results suggest that such heuristics may offer limited benefits over random sampling (Surkov et al., 2022). Beyond manual annotations or simple heuristics, CL variants differ in how they define difficulty and structure training. Teacher-student CL ranks samples via an external model (Xu et al., 2020; Soviany et al., 2022), while self-paced CL allows models to select samples based on their internal progress (Jiang et al.,

2015). Competence-based CL introduces a formal notion of model competence, and dynamically filters training samples (Platanios et al., 2019). Wu et al. (2021) examine whether curriculum or anti-curriculum ordering improves training, and find limited benefits over random sampling in standard settings. Beyond these mainstream variants, more recent work has extended curriculum learning into various specialized settings, including combining CL with active learning (Jafarpour et al., 2021), dual CL, which handles positive and negative samples separately (Zhu et al., 2022), and curriculum contrastive learning for knowledge graph entity typing (Wang et al., 2025). Recent work also applies curriculum learning to code language models by defining difficulty through static complexity measures (Naïr et al., 2024). Some methods follow curriculum principles without being explicitly framed as curriculum learning (Mindermann et al., 2022; Thakkar et al., 2023). In contrast to this line of work, we propose a CL framework relying on the difficulty predicted by the model itself, without relying on external models, metrics, or annotations.

2.3 Prompt-Based Fine-Tuning

Prompt-based Fine-tuning (PFT) has emerged as a powerful approach for adapting PLMs to downstream tasks, particularly in zero-shot and few-shot scenarios (Schick and Schütze, 2021a,c,b; Le Scao and Rush, 2021; Gao et al., 2021; Jin et al., 2022; An, 2023; Ma et al., 2023; Ullah et al., 2023; Xie and Li, 2024) . An important early stage of PFT research was marked by Pattern-Exploiting Training (PET), proposed by Schick and Schütze (2021c). Building on this, Schick and Schütze (2021a,b) further explored key factors such as prompt design, verbalizer selection, and self-training strategies, and extended PET to text generation tasks. In PFT, verbalizers can either be manually crafted or automatically optimized (Shin et al., 2020; Schick and Schütze, 2021a). Recent work has further extended PFT beyond monolingual settings to multilingual and cross-lingual tasks (Hu et al., 2022b; Ye et al., 2022; Wang et al., 2022; Ma et al., 2023). While early studies primarily focused on single-label classification, more recent efforts have adapted PFT to more complex settings such as multi-label classification (Yang et al., 2022). Recent work has also addressed semantic inconsistency and representation degeneration in prompt-based fine-tuning, proposing methods such as semantic consistency modeling (Xie and Li, 2024) and contrastive learning frameworks (Zhao et al., 2024a).

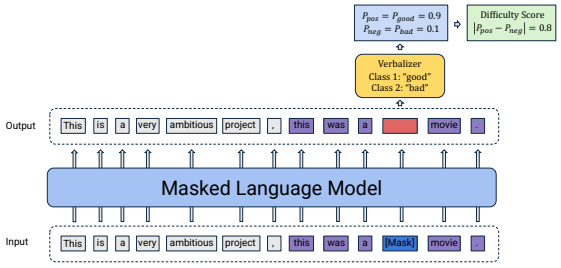

Figure 2: Illustration of the proposed difficulty scoring approach using masked language modeling and a verbalizer. The input sentence is processed to predict the masked token, and the resulting token probabilities are mapped to class labels through a verbalizer. In this example, the tokens "good" and "bad" represent the positive and negative classes, respectively. The difficulty score is then computed as the absolute difference between the class probabilities, reflecting the inherent complexity from the model's perspective.

3 Methodology

We propose a self-adaptive curriculum learning paradigm that relies on the difficulty predicted by the PLM itself. We use prompt templates (cf. §3.1) and the verbalizer component (cf. §3.2) to obtain the class probabilities, based on which we compute the difficulty score for each example (cf. §3.3). With the scores, we propose different sampling strategies for fine-tuning (cf. §3.4).

3.1 Prompt Construction

Our approach begins with the construction of task-specific prompts. The general structure is:

$$\texttt{Text} + \texttt{Template}$$

where `Text` is the actual text for which we want to obtain a prediction and `Template` is a few tokens that help the model to understand the task and make a prediction. `Template` always contains a special token `[MASK]`. We check the token distribution over vocabularies at the `[MASK]` position.

For example, in a sentiment analysis task for movie reviews, the prompt is formulated as: "This is a very ambitious project, this was a `[MASK]` movie.", where the first half, i.e., "This is a very ambitious project" is the actual sentence for classification while the rest is the template. Here, `[MASK]` prompts the model to predict an adjective token (e.g., *great*, *bad*), reflecting the sentiment of a "reviewer". The prompt templates we use for each downstream task are shown in §A.

3.2 Verbalizer Design

A verbalizer maps the token predicted at the `[MASK]` position to a task-specific category label. Taking binary classification for example, we define the verbalizer with carefully selected keywords aligned with the dataset and the task context:

$$V = \{\text{positive} \rightarrow \texttt{positive keyword}, \\ \text{negative} \rightarrow \texttt{negative keyword}\}$$

where positive/negative refer to the category, and `positive/negative keyword` are the tokens we use representing the corresponding category. Although multiple keywords per class can be considered, both previous research (Ma et al., 2023) and our preliminary results indicate that optimal performance is achieved when mapping each category to a single, clearly representative keyword. This verbalizer design is easily extendable to multi-class scenarios. We show our verbalizers in §A.

3.3 Difficulty Score Calculation

By feeding a prompt, we check the model's output logits at the `[MASK]` position. For each token w_i in the vocabulary $\mathbb{V}$, we obtain its corresponding logit z_i. We then calculate the probability of the token with the softmax function: $P(w_i) = \frac{e^{z_i}}{\sum_{w_j \in \mathbb{V}} e^{z_j}}$

Then, we extract the **label-specific probabilities** using verbalizers. Taking sentiment analysis (a **binary classification** task, for example, we compute the class probability by considering the selected keyword for each class:

$$P_{\text{pos}} = P(\texttt{positive keyword})$$
$$P_{\text{neg}} = P(\texttt{negative keyword})$$

Note that P_{pos} and P_{neg} are normalized so that $P_{\text{pos}} + P_{\text{neg}} = 1$. The difficulty score is then defined as the absolute difference between the two class probabilities: Difficulty Score $= |P_{\text{pos}} - P_{\text{neg}}|$.

Figure 2 illustrates the process of calculating the difficulty score. **The intuition is that a higher score indicates greater model confidence (lower difficulty), whereas a lower score suggests uncertainty (higher difficulty).** Our empirical results verify this intuition: Figure 1 shows that, even before training, examples with higher scores (less difficult) generally correspond to correct predictions. After training, the distribution shifts significantly toward higher scores (many examples become less difficult because the model has seen them), validating the effectiveness of our difficulty scoring method. This method easily generalizes to **multi-class classification** by defining difficulty score as the margin between the two highest class probabilities: Difficulty Score $= |P_{\text{max}} - P_{\text{second-max}}|$.

3.4 Sampling Strategies

Drawing inspiration from curriculum learning, we propose six sampling strategies grouped into three categories. The sampling relies on the difficulty score of each example. These strategies are designed to prioritize **"worth-learning"** examples during fine-tuning for specific tasks. Figure 3 presents an overview of our sampling strategies.

3.4.1 Naive Sequential Sampling

The most straightforward approach, akin to curriculum learning, is to arrange the training examples based on their difficulty scores and train the model using a fixed order. Let $X = \{x_n\}_{n=1}^{N}$ be the training examples, sorted by their associated difficulty scores s_n in **either ascending or descending order**. We propose two sampling strategies.

Easy to Difficult (E2D) Training examples are sorted **descendingly** according to the scores, such that $s_1 \geq s_2 \geq \cdots \geq s_n$, with x_1 being the easiest one and x_n the hardest one. Models are exposed to examples from x_1 to x_N sequentially.

Difficult to Easy (D2E) Training examples are sorted **ascendingly** according to the scores, such that $s_1 \leq s_2 \leq \cdots \leq s_n$, with x_1 being the hardest one and x_n the easiest one. Models are exposed to examples from x_1 to x_N sequentially.

3.4.2 Probability-Based Sampling

Our intuition is that sequentially exposing examples to the model can be overly rigid and lack diversity. This might result in the model's degradation in dealing with very easy or difficult examples. Therefore, we propose probability-based sampling strategies that introduce a more flexible and diverse training flow. Specifically, rather than following a fixed order, **examples are assigned probabilities based on their difficulty rankings**, enabling the model to encounter a controlled mixture of easy and hard examples. Given the ordered examples $X = [x_1, x_2, ..., x_N]$ according to their scores, the sampling probability for x_n is defined as:

$$P(x_n) = \frac{n^2}{\sum_{j=1}^{N} j^2}$$

That is, the sampling probability from x_1 to x_N increases. We propose two sampling strategies.

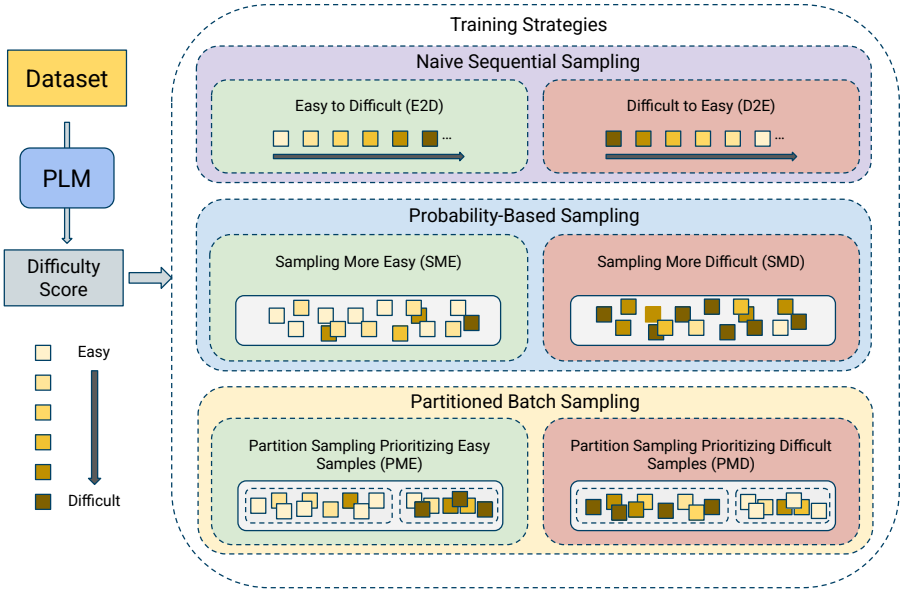

Figure 3: An illustration of our sampling strategies. Each example is associated with a difficulty score based on the PLM itself. Six sampling strategies are presented: **Naive Sequential Sampling** (E2D and D2E), **Probability-Based Sampling** (SME and SMD), and **Partitioned Batch Sampling** (PME and PMD). The difficulty of examples is indicated by color, with lighter colors representing easier samples and darker colors representing more difficult ones.

Sampling More Easy (SME) Training examples are sorted **ascendingly** according to the scores; thus, easier examples (higher ranks n) have larger probabilities of being sampled. This results in a sampling behavior in favor of easy examples with occasional difficult ones.

Sampling More Difficult (SMD) Training examples are sorted in **descending** order according to the scores; thus, more difficult examples (higher ranks n) have larger probabilities of being sampled. This results in a sampling behavior in favor of hard examples with occasional easy ones.

3.4.3 Partitioned Batch Sampling Strategies

As an extension of probability-based sampling, this method allows fine-grained control within each batch. Each batch B contains two partitions (B_1 and B_2) of examples, with **one partition focusing on sampling easier examples, while the other on more difficult ones**. Note that sampling within each partition is still based on the probability, rather than being deterministic. This also ensures diversity and avoids overfitting to a fixed progression. This approach enables a more dynamic and balanced mixture of easy and hard samples during fine-tuning. **We set $|B_1| > |B_2|$, aiming to give higher priority to partition B_1 during fine-tuning.**[2] We propose two sampling strategies.

Prioritizing Easy Samples (PME) The first partition B_1 prioritizes easy samples, while the second partition B_2 prioritizes difficult examples, achieved by assigning **two different probabilities to each example** x_n, one for B_1 and the other for B_2:

$$P_{B_1}(x_n) = \frac{n^2}{\sum_{j=1}^{N} j^2}, \quad P_{B_2}(x_n) = \frac{(N-n)^2}{\sum_{j=1}^{N} j^2}$$

In PME, the training examples are sorted in **ascending order** according to the scores. In this way, $P_{B_1}(x_n)$ prioritizes on easier examples while $P_{B_2}(x_n)$ prioritizes on harder examples.

Prioritizing Difficult Samples (PMD) Conversely, the training examples are sorted in **descending order** according to the scores. In this way, $P_{B_1}(x_n)$ prioritizes on harder examples while $P_{B_2}(x_n)$ prioritizes on easier examples.

4 Experimental Setup

We evaluate our proposed methods on four publicly available datasets, covering diverse NLP tasks to demonstrate the generality of our approach.

4.1 Datasets

Stanford Sentiment Treebank Binary (SST-2) SST-2 (Socher et al., 2013) is a balanced binary

[2] We set $|B_1| : |B_2| = 6 : 4$ based on preliminary results.

		SST-2				SST-5				HSOL				XNLI			
		Acc	F1	Prec	Rec	Acc	F1	Prec	Rec	Acc	F1	Prec	Rec	Acc	F1	Prec	Rec
BERT	Random	91.97	91.97	91.99	91.96	53.62	**52.37**	53.18	**52.05**	91.67	73.58	80.15	71.76	**84.01**	**84.02**	**84.23**	**84.01**
	Length	92.09	92.08	92.15	92.06	51.75	51.03	51.52	51.70	91.50	70.99	**80.30**	69.04	83.06	83.07	83.22	83.06
	E2D	92.39	92.39	92.39	92.41	52.16	47.12	**56.88**	48.17	90.95	70.20	76.21	70.50	82.83	82.87	83.52	82.83
	D2E	91.93	91.92	92.12	91.88	51.60	50.48	52.40	50.01	91.23	74.23	77.04	72.81	82.12	82.24	83.23	82.12
	SME	91.25	91.23	91.35	91.20	52.91	49.78	53.71	50.39	**91.81**	73.83	79.76	72.88	83.08	83.10	83.73	83.08
	SMD	91.48	91.47	91.53	91.45	52.73	50.92	51.84	51.14	91.51	74.71	79.21	72.22	82.31	82.41	83.28	82.31
	PME	91.40	91.38	91.59	91.35	**53.83**	50.72	54.33	50.40	91.67	74.46	79.19	73.05	83.75	83.78	84.02	83.75
	PMD	**92.62**	**92.61**	**92.73**	**92.60**	52.73	51.66	53.56	51.59	91.64	**76.14**	78.43	**74.76**	83.27	83.29	83.54	83.27
RoBERTa	Random	**94.11**	**94.11**	94.15	**94.10**	56.00	54.34	56.55	54.62	92.18	75.27	81.79	72.76	87.11	87.11	87.28	87.11
	Length	93.35	93.34	93.46	93.31	54.27	53.17	52.92	54.95	92.00	67.41	**85.02**	65.60	86.20	86.14	86.37	86.20
	E2D	93.92	93.92	**95.95**	93.91	57.00	53.29	56.64	53.76	90.96	73.98	77.04	74.38	85.73	85.76	86.23	85.73
	D2E	93.54	93.54	93.57	93.52	**57.07**	**55.30**	56.00	**55.70**	91.43	73.66	79.06	71.85	87.39	87.43	87.57	87.39
	SME	93.35	93.34	94.44	93.33	55.49	50.76	**57.76**	51.11	91.76	75.46	79.36	**75.79**	87.11	87.13	87.25	87.11
	SMD	93.39	93.37	93.56	93.34	56.46	53.83	56.50	53.51	91.57	75.23	78.14	74.09	86.86	86.96	87.42	86.86
	PME	93.85	93.84	93.89	93.82	55.76	52.17	57.13	52.64	92.14	**77.27**	80.05	75.64	86.86	86.91	87.17	86.86
	PMD	93.27	93.27	93.36	93.27	56.89	54.15	57.22	54.04	**92.53**	74.96	82.89	73.71	**87.47**	**87.49**	**87.58**	**87.47**

Table 1: Comparison of different sampling strategies and baselines across four datasets (SST-2, SST-5, HSOL, and XNLI) using BERT and RoBERTa as backbone models. Accuracy, F1 score, precision, and recall are reported. **Bold** (resp. underlined) entries highlight the best (resp. second-best) performance within each model group. For our proposed sampling approaches, we additionally use background colors red to indicate values higher than both baselines, blue to indicate values lower than both, and white to indicate performance between the two baselines. All results are averaged over runs with 3 different random seeds.

sentiment analysis dataset containing movie review sentences labeled as positive or negative.

Fine-grained Sentiment Analysis (SST-5) SST-5 dataset (Socher et al., 2013) contains sentences from movie reviews labeled into five fine-grained sentiment categories: very positive, positive, neutral, negative, and very negative.

Hate Speech Offensive Language (HSOL) The Hate Speech Offensive Language dataset (Davidson et al., 2017) includes tweets labeled into three categories: hate speech, offensive language, and neither, with a significant class imbalance.

Cross-lingual Natural Language Inference (XNLI) XNLI (Conneau et al., 2018) is a widely-used benchmark for natural language understanding tasks, providing sentence pairs labeled in three categories: entailment, neutral, or contradiction.

4.2 Models

We use `bert-base-uncased` (BERT-base) (Devlin et al., 2018) and `roberta-base` (RoBERTa-base) (Liu et al., 2019) as the base PLMs for all experiments. Since masked language modeling is the main objective in their pretraining, both models have a special `[MASK]` token in their vocabularies, which allows us to compute the difficulty score for each example in the training set of the downstream dataset and apply our sampling strategy for prompt-based fine-tuning, as introduced in §3.

4.3 Baselines

We consider two baselines: **Random** and **Length**. The **Random** baseline follows the classic strategy where a batch of training examples is randomly sampled from the training dataset. The **Length** baseline assumes that examples with more tokens are more difficult (Platanios et al., 2019). The examples are sorted from shortest to longest according to their tokenized length. **Length** not only reflects the inherent sentence length but also captures word rarity, as rare or uncommon words are typically tokenized into multiple subword units, thus resulting in longer sequences.

5 Results and Discussions

5.1 Main Result

Table 1 presents the accuracy, F1 score, precision, and recall scores on the test sets of the 4 datasets from the baselines and our training strategies.

RoBERTa consistently outperforms BERT across all datasets. RoBERTa shows overall better performance than BERT across all datasets under almost all sampling strategies, including Random and Length baseline. This is a strong indicator that RoBERTa's pretrained representation provides stronger generalization, especially under low-resource or imbalanced sampling conditions.

Random sampling is occasionally sufficient, but curriculum-sampling strategies offer more robust improvements. While the baseline Random

shows fair performance, especially in low-difficulty or well-balanced datasets (like SST-2), it gains inconsistent performance across harder datasets like SST-5 and XNLI. The baseline Length achieves slightly better performance than Random, indicating that curriculum learning with the sentence length as an indicator of difficulty works. However, the performance is also less consistent and usually worse than our proposed approaches. Our sampling strategies, especially PME, E2D, and SME, tend to offer more consistent gains, indicating the effectiveness of using the model's own prediction for difficulty calculation of training examples.

PMD achieves the highest performance in most cases. The PMD strategy yields top performance (highlighted in bold) on multiple datasets for both BERT and RoBERTa, especially on SST-2 and XNLI. Its consistent superiority suggests that its dynamic sampling mechanism effectively emphasizes worth-learning examples during training.

Dataset difficulty affects the benefit of sampling strategies. On easier datasets such as SST-2 and HSOL, most strategies achieve high and stable results, and the performance gap between baselines and sampling-based methods remains relatively small. In contrast, on more challenging datasets like SST-5 and XNLI, the performance differences are more pronounced, indicating that sampling strategies provide greater benefits when the task involves finer-grained classes.

On imbalanced datasets, the proposed sampling strategies offer clear advantages. In datasets like HSOL, which exhibit label imbalance or fine-grained distinctions, our sampling strategies, such as PME and SME, consistently achieve higher F1 scores compared to baselines. This indicates their effectiveness in promoting better representation of minority or harder-to-learn classes, improving the overall balance between precision and recall.

5.2 Training Progression Analysis

To further understand the benefit of our methods, we analyze the changes in accuracy and loss on the validation set for each dataset within a single epoch of fine-tuning. Throughout the epoch, we store a checkpoint every 10% of the training samples. We then evaluate each checkpoint on the validation set. Consequently, we save the average accuracy and loss on the validation set at 10 different checkpoints. We discuss the trend of accuracy of SST-2 and SST-5 in the following. The complete results (accuracy and loss) for each dataset are presented in §C.

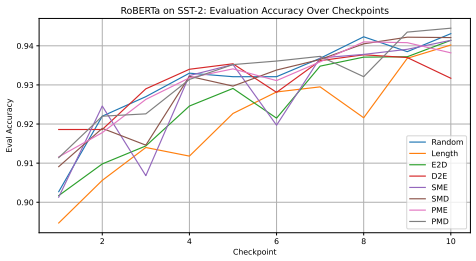

Figure 4: Progression of accuracy during a single epoch on **SST-2**. Each checkpoint corresponds to a model seeing 10% of the training examples.

Figure 4 presents the RoBERTa results on **SST-2**. At the first checkpoint, sampling strategies D2E, PME, PMD, and SMD show a clear advantage, far exceeding both the baselines and the E2D and SME strategies. This might indicate that early exposure to difficult examples might be helpful. Throughout training, all methods exhibit some degree of fluctuation. At the final checkpoint, most methods, including the baselines, continue to improve. This suggests that, despite fluctuations during training, most methods benefit from longer training time.

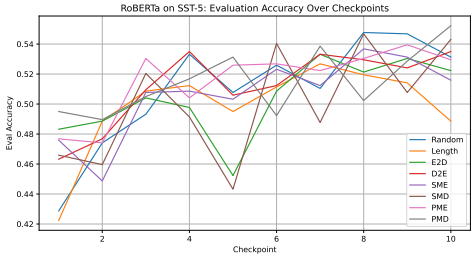

Figure 5: Progression of accuracy during a single epoch on **SST-5**. Each checkpoint corresponds to a model seeing 10% of the training examples.

Figure 5 presents the RoBERTa results on **SST-5**. Different from the trend from SST-2, we observe that all our training strategies significantly outperform the baseline at the first checkpoint, indicating that, compared to Random and Length, our methods enable RoBERTa to learn useful features more rapidly in the early stages. However, almost all strategies exhibit substantial fluctuations throughout training. In the final phase, PMD, SMD, and D2E still show improvements, while other strategies decline. Among them, PMD achieves the highest performance through a rapid increase. This might suggest that, for multi-class classification,

		SST-2				SST-5				HSOL				XNLI			
		Acc	F1	Prec	Rec	Acc	F1	Prec	Rec	Acc	F1	Prec	Rec	Acc	F1	Prec	Rec
BERT	Random	80.85	80.60	81.92	80.78	36.58	31.22	38.43	34.07	77.69	32.58	**47.70**	34.96	35.52	33.41	36.07	35.52
	Length	68.12	65.18	75.96	67.62	37.36	32.68	35.87	**35.19**	77.29	**39.01**	42.80	**39.05**	35.07	25.99	**37.38**	35.07
	E2D	52.29	36.97	72.21	51.41	30.94	24.97	29.59	33.08	76.63	37.56	45.87	37.47	34.32	30.89	35.33	34.32
	D2E	**84.02**	**84.01**	**84.05**	**84.01**	39.23	24.28	**45.59**	31.31	76.28	34.83	41.11	35.89	33.73	27.70	35.31	33.73
	SME	81.00	80.94	81.34	80.99	**40.44**	30.22	44.68	33.88	77.07	37.34	45.87	37.66	35.24	31.68	35.77	35.24
	SMD	78.36	78.13	79.36	78.35	**40.44**	29.60	39.47	33.44	77.28	34.42	47.53	36.05	35.32	29.17	36.04	35.32
	PME	79.70	79.49	80.71	78.35	39.67	32.39	37.98	34.42	77.89	35.22	47.04	36.48	36.10	35.20	36.21	36.10
	PMD	79.32	78.87	81.38	79.31	40.62	32.91	38.29	35.13	**77.92**	34.91	45.60	36.44	**36.39**	**35.69**	36.40	**36.39**
RoBERTa	Random	89.64	89.63	89.75	89.66	45.11	34.54	44.46	38.21	**80.67**	**42.44**	53.47	41.42	35.53	**31.59**	33.70	35.53
	Length	84.02	83.77	85.39	83.85	40.77	28.69	40.64	33.25	79.35	39.33	50.57	39.22	35.26	27.62	26.37	35.26
	E2D	87.99	87.99	88.02	87.99	43.18	**35.35**	39.61	37.51	77.20	35.50	50.24	36.20	35.55	29.81	34.73	35.55
	D2E	90.56	90.55	90.58	90.54	45.10	26.26	34.92	35.65	77.69	38.49	47.48	38.63	32.48	22.15	32.98	32.48
	SME	90.37	90.36	90.42	90.34	45.69	34.29	39.87	**38.32**	78.62	34.20	**56.91**	36.01	**35.61**	29.38	34.86	**35.61**
	SMD	**90.86**	**90.86**	**90.87**	**90.86**	43.79	28.46	37.17	35.63	78.68	35.17	54.04	36.62	33.27	28.36	34.86	33.27
	PME	88.95	88.93	89.17	88.93	**47.41**	31.39	44.86	38.24	80.64	42.04	53.97	**41.76**	34.29	30.90	**35.02**	34.29
	PMD	90.06	90.03	90.35	90.01	45.13	32.33	**45.13**	37.11	79.99	41.34	52.05	40.84	33.83	31.13	34.57	33.83

Table 2: Comparison of different sampling strategies and baselines across four datasets (SST-2, SST-5, HSOL, and XNLI) under **few-shot learning** setting with 64 training instances. Accuracy, F1 score, precision, and recall are reported. **Bold** (resp. underlined) entries highlight the best (resp. second-best) performance within each model group. For our proposed sampling approaches, we additionally use background colors red to indicate values higher than both baselines, blue to indicate values lower than both, and white to indicate performance between the two baselines. All results are averaged over runs with 3 different random seeds.

prioritizing difficult samples can facilitate more stable learning in the last stage of training.

5.3 Few-Shot Learning

To further investigate the benefit of our strategies under the scenarios where limited training data are present, we conduct a few-shot learning evaluation, similar to the setup of Ma et al. (2023), using the 4 datasets. Specifically, we select the **top 64** ranked examples in each sampling strategy.[3] The number of 64 samples is chosen to ensure sufficient diversity across difficulty levels. The PLMs are trained on these examples solely, and Table 2 presents the results of the resulting models on the test set.

RoBERTa shows a clear advantage over BERT, especially on SST-2 and SST-5. Similar to the results shown in Table 1, RoBERTa also achieves better performance than BERT. We even notice that the performance on SST-2 is already close to the fully supervised performance reported in Table 1. For HSOL and XNLI, however, the gap between the two models is much smaller. We assume this is due to dataset imbalance and difficulty, which limit the effectiveness of few-shot learning.

On SST-2 and SST-5, most of our sampling strategies consistently outperform both baselines except for E2D. Length performs noticeably worse than the other methods, which is because only short-length examples are exposed to the model. On the other hand, the baseline Random remains relatively strong, as it sees both short and long examples. We notice that E2D in BERT fails to train the model properly, which is expected since the model only sees easy examples on which the model should already perform very well, even without any fine-tuning. For other training strategies, we generally see improvements. Strategies such as D2E and probability-based methods like SME, PME, and PMD show substantial improvements across multiple metrics, indicating that hard examples are particularly important in few-shot learning.

For the more challenging inference dataset XNLI, using only 64 samples appears insufficient for training. We notice that all models obtain much lower performance in XNLI compared with the results of full-dataset training (cf. Table 1). This indicates the difficulty of XNLI dataset – only when enough training instances are available, the model can learn the necessary features for making reasonable decisions. As a result, based on the poor performance, it is difficult to draw clear conclusions regarding which sampling strategy is more effective on XNLI. We hypothesize that increasing the number of training samples, e.g., 128 or 256, could alleviate the problem.

6 Conclusion

In this work, we introduced a self-adaptive curriculum learning paradigm that leverages a PLM's own

[3]We use the top 64 ranked examples for all strategies except Random, for which examples are randomly sampled.

confidence to estimate the difficulty of training examples. We further propose a range of sampling strategies: sequential, probabilistic, and partitioned, and verify the effectiveness on multiple NLU tasks. Our empirical results show improved performance in both full-data and few-shot settings, confirming the utility of model-predicted difficulty as a training signal. This paradigm offers a scalable and model-centric alternative to traditional curriculum learning, offering insights for broader applications across diverse NLU tasks.

Limitations

We propose a self-adaptive curriculum learning paradigm that relies on the difficulty score predicted by the model itself. Despite promising results, several limitations remain, particularly related to GPU memory constraints, which restrict input size and dataset coverage. With access to more powerful GPUs, we could conduct experiments on larger and more comprehensive datasets. We compare with representative baselines: **Random** and **Length**. Future work can also consider other difficulty-based alternatives, such as rarity- or attention-based sampling. Furthermore, our current experiments are limited to English classification tasks; future work should explore the applicability of our method to multilingual and cross-lingual settings.

Our current implementation is based on single-token classification settings. Extending difficulty scoring to multi-token or generative tasks (e.g., QA, summarization) remains an open direction. Furthermore, since prompt-based learning is highly sensitive to prompt design, experimenting with different templates and verbalizer words could further enhance model performance and interpretability. Another possible limitation is the lack of direct comparison with human-annotated difficulty levels, which could offer further insight into the alignment or divergence between model-based and human intuition.

Addressing imbalanced datasets by integrating dual curriculum learning concepts and implementing dynamic or multi-phase training strategies could also improve adaptability and efficiency. Overcoming these challenges would significantly boost the effectiveness and generalizability of our sampling strategies.

Acknowledgments

This work was funded by Deutsche Forschungsgemeinschaft (project SCHU 2246/14-1).

References

Bo An. 2023. Prompt-based for low-resource tibetan text classification. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, 22(8).

Abdul Hameed Azeemi, Ihsan Ayyub Qazi, and Agha Ali Raza. 2025. To label or not to label: Hybrid active learning for neural machine translation. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 3071–3082, Abu Dhabi, UAE. Association for Computational Linguistics.

Guangji Bai, Zheng Chai, Chen Ling, Shiyu Wang, Jiaying Lu, Nan Zhang, Tingwei Shi, Ziyang Yu, Mengdan Zhu, Yifei Zhang, Xinyuan Song, Carl Yang, Yue Cheng, and Liang Zhao. 2024. Beyond efficiency: A systematic survey of resource-efficient large language models. *Preprint*, arXiv:2401.00625.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009, Montreal, Quebec, Canada, June 14-18, 2009*, volume 382 of *ACM International Conference Proceeding Series*, pages 41–48. ACM.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, Wei Ye, Yue Zhang, Yi Chang, Philip S. Yu, Qiang Yang, and Xing Xie. 2024. A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology*, 15(3).

Jiaao Chen, Dinghan Shen, Weizhu Chen, and Diyi Yang. 2021. HiddenCut: Simple data augmentation for natural language understanding with better generalizability. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4380–4390, Online. Association for Computational Linguistics.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel R. Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Sophia R. Cunningham, Dominique Archambault, and Austin Kung. 2024. Efficient training and inference: Techniques for large language models using llama.

Kunal Dahiya, Nilesh Gupta, Deepak Saini, Akshay Soni, Yajun Wang, Kushal Dave, Jian Jiao, Gururaj K, Prasenjit Dey, Amit Singh, and 1 others. 2023. Ngame: Negative mining-aware mini-batching for extreme classification. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, pages 258–266.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. In *Proceedings of the 11th International AAAI Conference on Web and Social Media*, ICWSM '17, pages 512–515.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, Jing Yi, Weilin Zhao, Xiaozhi Wang, Zhiyuan Liu, Hai-Tao Zheng, Jianfei Chen, Yang Liu, Jie Tang, Juanzi Li, and Maosong Sun. 2023. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 5(3):220–235.

Bonaventure F. P. Dossou, Atnafu Lambebo Tonja, Oreen Yousuf, Salomey Osei, Abigail Oppong, Iyanuoluwa Shode, Oluwabusayo Olufunke Awoyomi, and Chris Emezue. 2022. AfroLM: A self-active learning-based multilingual pretrained language model for 23 African languages. In *Proceedings of the Third Workshop on Simple and Efficient Natural Language Processing (SustaiNLP)*, pages 52–64, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Jingfei Du, Edouard Grave, Beliz Gunel, Vishrav Chaudhary, Onur Celebi, Michael Auli, Veselin Stoyanov, and Alexis Conneau. 2021. Self-training improves pre-training for natural language understanding. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5408–5418, Online. Association for Computational Linguistics.

Shen Gao, Zhengliang Shi, Minghang Zhu, Bowen Fang, Xin Xin, Pengjie Ren, Zhumin Chen, Jun Ma, and Zhaochun Ren. 2024. Confucius: Iterative tool learning from introspection feedback by easy-to-difficult curriculum. In *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*, pages 18030–18038. AAAI Press.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830, Online. Association for Computational Linguistics.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022a. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Linmei Hu, Zeyi Liu, Ziwang Zhao, Lei Hou, Liqiang Nie, and Juanzi Li. 2024. A survey of knowledge enhanced pre-trained language models. *IEEE Transactions on Knowledge and Data Engineering*, 36(4):1413–1430.

Shengding Hu, Ning Ding, Huadong Wang, Zhiyuan Liu, Jingang Wang, Juanzi Li, Wei Wu, and Maosong Sun. 2022b. Knowledgeable prompt-tuning: Incorporating knowledge into prompt verbalizer for text classification. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2225–2240, Dublin, Ireland. Association for Computational Linguistics.

Jianheng Huang, Leyang Cui, Ante Wang, Chengyi Yang, Xinting Liao, Linfeng Song, Junfeng Yao, and Jinsong Su. 2024. Mitigating catastrophic forgetting in large language models with self-synthesized rehearsal. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1416–1428, Bangkok, Thailand. Association for Computational Linguistics.

Borna Jafarpour, Dawn Sepehr, and Nick Pogrebnyakov. 2021. Active curriculum learning. In *Proceedings of the First Workshop on Interactive Learning for Natural Language Processing*, pages 40–45, Online. Association for Computational Linguistics.

Lu Jiang, Deyu Meng, Qian Zhao, Shiguang Shan, and Alexander Hauptmann. 2015. Self-paced curriculum learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29.

Woojeong Jin, Yu Cheng, Yelong Shen, Weizhu Chen, and Xiang Ren. 2022. A good prompt is worth millions of parameters: Low-resource prompt-based learning for vision-language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2763–2775, Dublin, Ireland. Association for Computational Linguistics.

Zixuan Ke, Bing Liu, Nianzu Ma, Hu Xu, and Lei Shu. 2021. Achieving forgetting prevention and knowledge transfer in continual learning. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 22443–22456.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526.

Teven Le Scao and Alexander Rush. 2021. How many data points is a prompt worth? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2627–2636, Online. Association for Computational Linguistics.

David D. Lewis and William A. Gale. 1994. A sequential algorithm for training text classifiers. *CoRR*, abs/cmp-lg/9407020.

Jia Li, Chongyang Tao, Wei Wu, Yansong Feng, Dongyan Zhao, and Rui Yan. 2019. Sampling matters! an empirical study of negative sampling strategies for learning of matching models in retrieval-based dialogue systems. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1291–1296, Hong Kong, China. Association for Computational Linguistics.

Yihong Liu, Haotian Ye, Chunlan Ma, Mingyang Wang, and Hinrich Schütze. 2024. Langsamp: Language-script aware multilingual pretraining. *Preprint*, arXiv:2409.18199.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *Preprint*, arXiv:1907.11692.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Bolei Ma, Ercong Nie, Helmut Schmid, and Hinrich Schuetze. 2023. Is prompt-based finetuning always better than vanilla finetuning? insights from cross-lingual language understanding. In *Proceedings of the 19th Conference on Natural Language Processing (KONVENS 2023)*, pages 1–16, Ingolstadt, Germany. Association for Computational Lingustics.

Adyasha Maharana and Mohit Bansal. 2022. On curriculum learning for commonsense reasoning. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 983–992, Seattle, United States. Association for Computational Linguistics.

Sören Mindermann, Jan Brauner, Muhammed Razzak, Mrinank Sharma, Andreas Kirsch, Winnie Xu, Benedikt Höltgen, Aidan N. Gomez, Adrien Morisot, Sebastian Farquhar, and Yarin Gal. 2022. Prioritized training on points that are learnable, worth learning, and not yet learnt. *Preprint*, arXiv:2206.07137.

Moin Nadeem, Tianxing He, Kyunghyun Cho, and James Glass. 2020. A systematic characterization of sampling algorithms for open-ended language generation. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 334–346, Suzhou, China. Association for Computational Linguistics.

Marwa Naïr, Kamel Yamani, Lynda Lhadj, and Riyadh Baghdadi. 2024. Curriculum learning for small code language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 4: Student Research Workshop)*, pages 390–401, Bangkok, Thailand. Association for Computational Linguistics.

Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian. 2024. A comprehensive overview of large language models. *Preprint*, arXiv:2307.06435.

Jerzy Neyman. 1934. On the two different aspects of the representative method: The method of stratified sampling and the method of purposive selection. *Journal of the Royal Statistical Society*, 97(4):558–625.

Emmanouil Antonios Platanios, Otilia Stretcu, Graham Neubig, Barnabas Poczos, and Tom Mitchell. 2019. Competence-based curriculum learning for neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1162–1172, Minneapolis, Minnesota. Association for Computational Linguistics.

Longhua Qian, Guodong Zhou, Fang Kong, and Qiaoming Zhu. 2009. Semi-supervised learning for semantic relation classification using stratified sampling strategy. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1437–1445, Singapore. Association for Computational Linguistics.

Leonardo Ranaldi, Giulia Pucci, and Fabio Massimo Zanzotto. 2023. Modeling easiness for training transformers with curriculum learning. In *Proceedings of the 14th International Conference on Recent Advances in Natural Language Processing*, pages 937–948, Varna, Bulgaria. INCOMA Ltd., Shoumen, Bulgaria.

Timo Schick and Hinrich Schütze. 2021a. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, Online. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021b. Few-shot text generation with natural language instructions. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 390–402, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021c. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2339–2352, Online. Association for Computational Linguistics.

Yunfan Shao, Linyang Li, Zhaoye Fei, Hang Yan, Dahua Lin, and Xipeng Qiu. 2024. Balanced data sampling for language model training with clustering. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 14012–14023, Bangkok, Thailand. Association for Computational Linguistics.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Linxin Song, Jieyu Zhang, Tianxiang Yang, and Masayuki Goto. 2022. Adaptive ranking-based sample selection for weakly supervised class-imbalanced text classification. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 1641–1655, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Petru Soviany, Radu Tudor Ionescu, Paolo Rota, and Nicu Sebe. 2022. Curriculum learning: A survey. *Preprint*, arXiv:2101.10382.

Maxim Surkov, Vladislav Mosin, and Ivan P. Yamshchikov. 2022. Do data-based curricula work? In *Proceedings of the Third Workshop on Insights from Negative Results in NLP*, pages 119–128, Dublin, Ireland. Association for Computational Linguistics.

Qingyu Tan, Lu Xu, Lidong Bing, and Hwee Tou Ng. 2023. Class-adaptive self-training for relation extraction with incompletely annotated training data. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 8630–8643, Toronto, Canada. Association for Computational Linguistics.

Megh Thakkar, Tolga Bolukbasi, Sriram Ganapathy, Shikhar Vashishth, Sarath Chandar, and Partha Talukdar. 2023. Self-influence guided data reweighting for language model pre-training. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2033–2045, Singapore. Association for Computational Linguistics.

Faizad Ullah, Ubaid Azam, Ali Faheem, Faisal Kamiran, and Asim Karim. 2023. Comparing prompt-based and standard fine-tuning for Urdu text classification. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 6747–6754, Singapore. Association for Computational Linguistics.

Han Wang, Canwen Xu, and Julian McAuley. 2022. Automatic multi-label prompting: Simple and interpretable few-shot classification. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5483–5492, Seattle, United States. Association for Computational Linguistics.

Hao Wang, Minghua Nuo, and Shan Jiang. 2025. Knowledge graph entity typing with curriculum contrastive learning. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 574–583, Abu Dhabi, UAE. Association for Computational Linguistics.

Xiaoxia Wu, Ethan Dyer, and Behnam Neyshabur. 2021. When do curricula work? In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Zhipeng Xie and Yahe Li. 2024. Discriminative language model as semantic consistency scorer for prompt-based few-shot text classification. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 4968–4977, Torino, Italia. ELRA and ICCL.

Benfeng Xu, Licheng Zhang, Zhendong Mao, Quan Wang, Hongtao Xie, and Yongdong Zhang. 2020. Curriculum learning for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6095–6104, Online. Association for Computational Linguistics.

Zhichao Yang, Shufan Wang, Bhanu Pratap Singh Rawat, Avijit Mitra, and Hong Yu. 2022. Knowledge injected prompt based fine-tuning for multi-label few-shot ICD coding. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 1767–1781, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Hongbin Ye, Ningyu Zhang, Shumin Deng, Xiang Chen, Hui Chen, Feiyu Xiong, Xi Chen, and Huajun Chen. 2022. Ontology-enhanced prompt-tuning for few-shot learning. In *Proceedings of the ACM Web Conference 2022*, WWW '22. ACM.

Donghan Yu, Chenguang Zhu, Yiming Yang, and Michael Zeng. 2022a. JAKET: joint pre-training of knowledge graph and language understanding. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 11630–11638. AAAI Press.

Yue Yu, Lingkai Kong, Jieyu Zhang, Rongzhi Zhang, and Chao Zhang. 2022b. Actune: Uncertainty-based active self-training for active fine-tuning of pretrained language models. In *Proceedings of the 2022 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 1422–1436.

Michelle Yuan, Hsuan-Tien Lin, and Jordan Boyd-Graber. 2020. Cold-start active learning through self-supervised language modeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7935–7948, Online. Association for Computational Linguistics.

Zhisong Zhang, Emma Strubell, and Eduard Hovy. 2022. A survey of active learning for natural language processing. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6166–6190, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Qingyan Zhao, Ruifang He, Jinpeng Zhang, Chang Liu, and Bo Wang. 2024a. Representation degeneration problem in prompt-based models for natural language understanding. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 13946–13957, Torino, Italia. ELRA and ICCL.

Raoyuan Zhao, Abdullatif Köksal, Yihong Liu, Leonie Weissweiler, Anna Korhonen, and Hinrich Schuetze. 2024b. SynthEval: Hybrid behavioral testing of NLP models with synthetic CheckLists. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 7017–7034, Miami, Florida, USA. Association for Computational Linguistics.

Qingqing Zhu, Xiuying Chen, Pengfei Wu, JunFei Liu, and Dongyan Zhao. 2021. Combining curriculum learning and knowledge distillation for dialogue generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1284–1295, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yutao Zhu, Jian-Yun Nie, Yixuan Su, Haonan Chen, Xinyu Zhang, and Zhicheng Dou. 2022. From easy to hard: A dual curriculum learning framework for context-aware document ranking. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management, Atlanta, GA, USA, October 17-21, 2022*, pages 2784–2794. ACM.

A Training Details

We evaluate our proposed methods on four publicly available datasets, covering diverse NLP tasks to demonstrate the generality of our approach. Below we describe each dataset, including preprocessing, prompt templates, and verbalizer definitions.

A.1 Stanford Sentiment Treebank Binary (SST-2)

We randomly partition the original training set into training (80%) and validation sets (20%), maintaining label distribution. The original validation set serves as our test set. Tokenized samples are truncated at 128 tokens. The prompt template and verbalizer are set as follows:

$$x + \text{“this was a [MASK] movie.”}$$

$$V = \{\text{positive} \rightarrow \text{“great”},\ \text{negative} \rightarrow \text{“bad”}\}$$

A.2 Fine-grained Sentiment Analysis (SST-5)

The maximum token length is set to 128 tokens. The prompt template and verbalizer are set as follows:

$$x + \text{“this was a [MASK] movie.”}$$

$$V = \left\{ \begin{array}{l} \text{very positive} \rightarrow \text{“amazing”}, \\ \text{positive} \rightarrow \text{“great”}, \\ \text{neutral} \rightarrow \text{“okay”}, \\ \text{negative} \rightarrow \text{“bad”}, \\ \text{very negative} \rightarrow \text{“terrible”} \end{array} \right\}$$

A.3 Hate Speech Offensive Language (HSOL)

We split the original dataset into training (80%), validation (10%), and test (10%) subsets, maintaining class distribution. Maximum token length is limited to 128 tokens. The prompt template and verbalizer are set as follows:

$$x + \text{“this was [MASK].”}$$

$$V = \left\{ \begin{array}{l} \text{hate speech} \rightarrow \text{“hateful”}, \\ \text{offensive} \rightarrow \text{“offensive”}, \\ \text{neither} \rightarrow \text{“neutral”} \end{array} \right\}$$

A.4 Cross-lingual Natural Language Inference (XNLI)

We limit maximum sequence length to 128 tokens. The prompt template and verbalizer are set as follows:

$$\begin{array}{l} \text{Sentence 1 is \{premise\},} \\ \text{sentence 2 is \{hypothesis\}.} \\ \text{They are [MASK].} \end{array}$$

$$V = \left\{ \begin{array}{l} \text{entailment} \rightarrow \text{“entailed”}, \\ \text{neutral} \rightarrow \text{“neutral”}, \\ \text{contradiction} \rightarrow \text{“contradictory”} \end{array} \right\}$$

A.5 Hyperparameter Settings

Hyperparameters are carefully tuned through empirical tests for optimal performance and computational efficiency. Based on preliminary experiments, we set the learning rate to 1×10^{-5}, batch size to 16 for all experiments. For the main experiment and few-shot task, each model is trained for 5 epochs. For detailed analysis we only train the model for 1 epoch. The optimizer used is AdamW (Loshchilov and Hutter, 2017) coupled with a linear scheduler (no warm-up steps).

For partition sampling strategies (PME and PMD), we set the batch partitions in a 6:4 ratio (9 samples in the first partition and 7 samples in the second).

Model selection for evaluation on the test set is based on the highest validation accuracy achieved during training.

During training, we maintain the same hyperparameters across all six sampling strategies and three experimental setups to ensure consistency in comparison. To mitigate the impact of random variation, we conduct each experiment using three different random seeds {66, 88, 99} and report the averaged results. For detailed analysis we use the result of seed 66. All experiments are conducted using NVIDIA GeForce GTX 1080 Ti GPUs with 11 GB of memory. The entire pipeline is implemented using the PyTorch framework, which facilitated efficient training and evaluation.

B Reproducibility

The code for data processing and model training is available at the following Github repository: https://github.com/alitanokiki/self-adaptive-curriculum-nlu-acl2025.

C Detailed Analysis

This section presents the results of all detailed analyses that were not included in the main text.

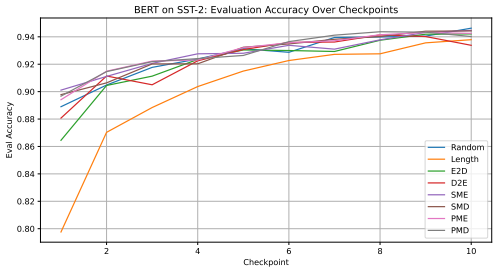

Figure 6: Average evaluation accuracy on BERT recorded at 10 checkpoints during a single epoch on SST-2.

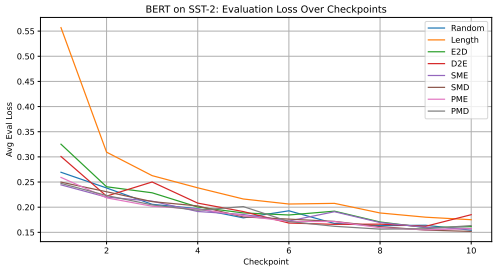

Figure 7: Average evaluation loss on BERT recorded at 10 checkpoints during a single epoch on SST-2.

As shown in Figure 6 and 7, probabilistic sampling methods (SME, SMD, PME, PMD) generally perform better.

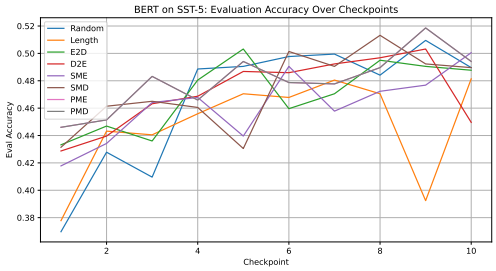

Figure 8: Average evaluation accuracy on BERT recorded at 10 checkpoints during a single epoch on SST-5.

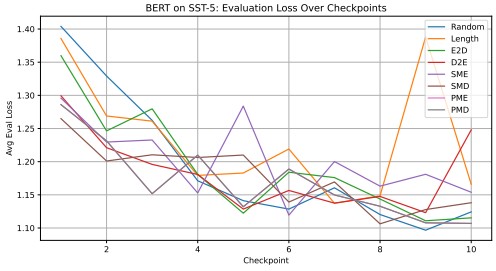

Figure 9: Average evaluation loss on BERT recorded at 10 checkpoints during a single epoch on SST-5.

Figure 8 shows that all our training strategies start with strong performance. Performance fluctuates across strategies, with D2E performing significantly worse at the end. According to Figure 9, SME achieves high accuracy but also results in higher loss.

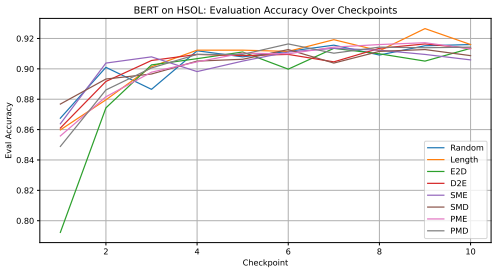

Figure 10: Average evaluation accuracy on BERT recorded at 10 checkpoints during a single epoch on HSOL.

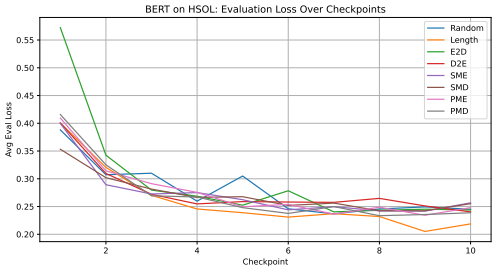

Figure 11: Average evaluation loss on BERT recorded at 10 checkpoints during a single epoch on HSOL.

Figure 10 and 11 indicate that E2D performs poorly at the beginning on imbalanced datasets. It is evident that after one epoch, our strategies no longer outperform the two baselines.

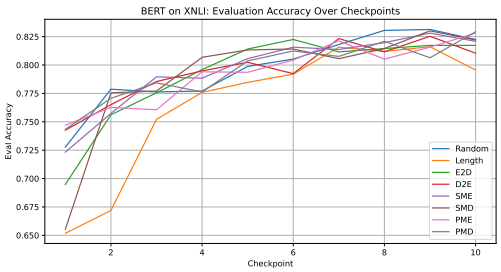

Figure 12: Average evaluation accuracy on BERT recorded at 10 checkpoints during a single epoch on XNLI.

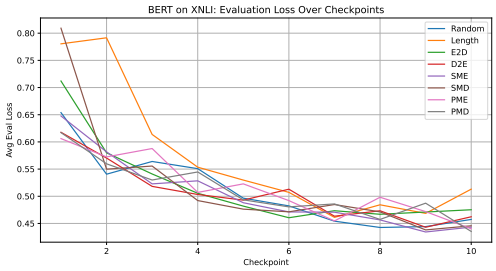

Figure 13: Average evaluation loss on BERT recorded at 10 checkpoints during a single epoch on XNLI.

As shown in Figure 12 and 13, SMD starts off weaker but converges quickly. All probabilistic sampling methods (SME, SMD, PME, PMD) perform well in the end.

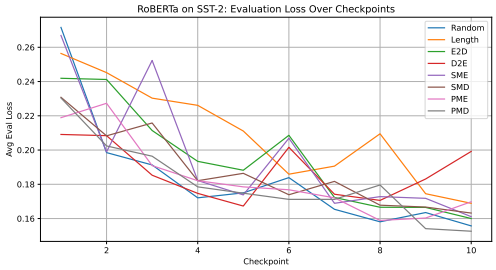

Figure 14: Average evaluation loss on RoBERTa recorded at 10 checkpoints during a single epoch on SST-2.

From Figure 14, we see that D2E has low initial loss, but ends with the highest loss after one epoch.

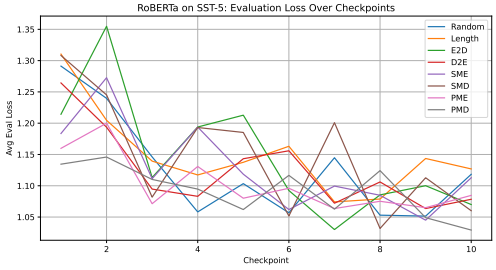

Figure 15: Average evaluation loss on RoBERTa recorded at 10 checkpoints during a single epoch on SST-5.

As shown in Figure 15, PMD maintains the lowest and most stable loss throughout training.

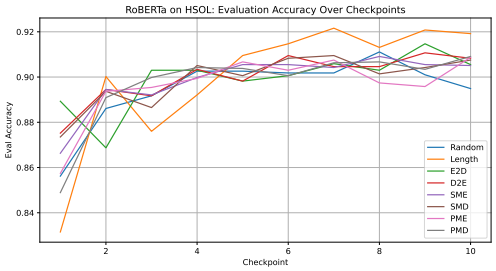

Figure 16: Average evaluation accuracy on RoBERTa recorded at 10 checkpoints during a single epoch on HSOL.

Figure 16 reveals that E2D shows early advantages, but the Length baseline performs best in the final stage.

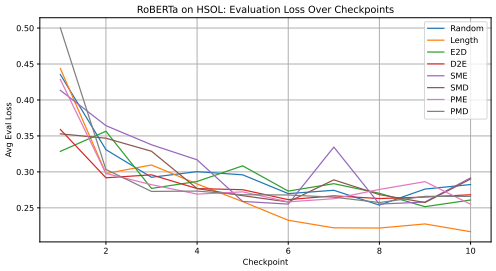

Figure 17: Average evaluation loss on RoBERTa recorded at 10 checkpoints during a single epoch on HSOL.

According to Figure 17, PMD initially has the highest loss, but it decreases rapidly.

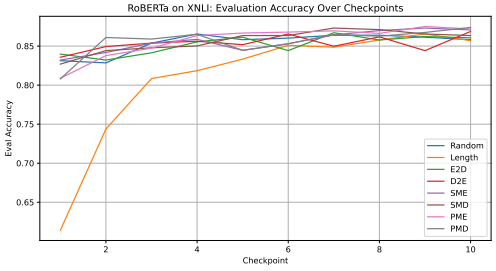

Figure 18: Average evaluation accuracy on RoBERTa recorded at 10 checkpoints during a single epoch on XNLI.

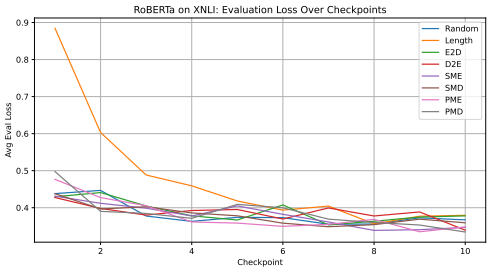

Figure 19: Average evaluation loss on RoBERTa recorded at 10 checkpoints during a single epoch on XNLI.

Figure 18 and 19 show that apart from the baseline Length, differences in performance across methods are minor.

D Difficulty Score Distribution Over Training Time

We analyze the evolution of sample difficulty score distributions under various training strategies across different datasets, using both BERT and RoBERTa models. While different strategies exhibit similar trends within the same dataset, the distributional patterns vary notably across datasets. Due to the consistency observed within each dataset, we take the BERT model as a representative example to illustrate these trends. Specifically, we present the score distribution changes of BERT trained with the baseline Random on each dataset, highlighting how dataset characteristics influence learning dynamics.

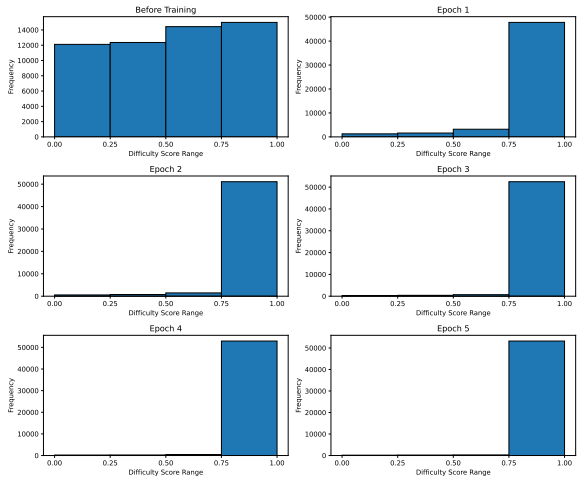

Figure 20: Sample difficulty score distributions on SST-2 before training and after each of five training epochs using BERT.

As shown in Figure 20, the initial difficulty score

distribution on the SST-2 dataset is relatively uniform. After the first epoch, the number of easy samples increases sharply, indicating that the model has learned substantially during the initial phase. The shift toward higher scores suggests increased model confidence. In subsequent epochs, the distribution stabilizes, reflecting more consistent learning dynamics.

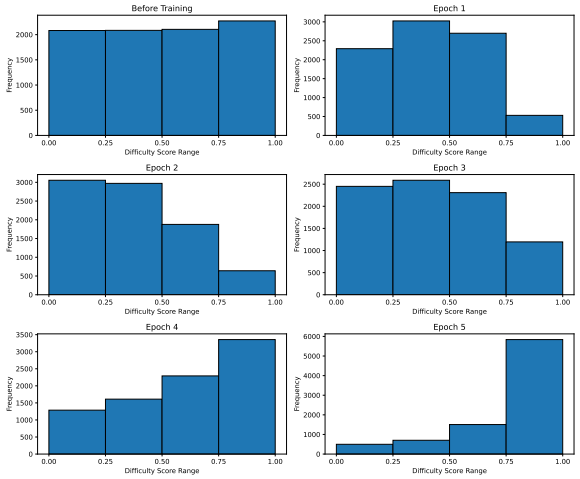

Figure 21: Sample difficulty score distributions on SST-5 before training and after each of five training epochs using BERT.

Figure 21 shows the evolution of difficulty score distribution for the BERT model on the SST-5 dataset. After one epoch, the number of relatively difficult samples increases, which may be attributed to the way difficulty scores are computed. One possible explanation is that, for multi-class classification, the difficulty score is defined as the absolute difference between the top two class probabilities. In this dataset, certain samples may have high but very close probabilities for adjacent sentiment classes, such as "negative" and "very negative" or "positive" and "very positive." As the model begins to learn useful features, the score difference of these low-confidence difficult samples tends to increase. Once the model has acquired more discriminative features, it becomes easier to correctly classify these borderline cases, resulting in higher overall accuracy. In this sense, low-confidence difficult samples may be the easiest to convert from incorrect to correct predictions. This interpretation is further supported by the observed score distribution, indicating that the model learned meaningful features within the first epoch.

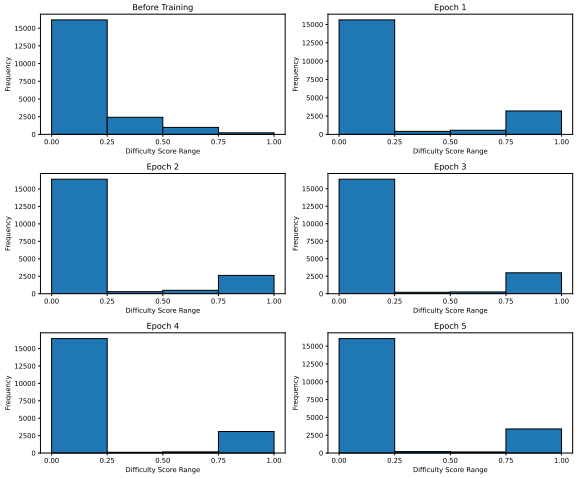

Figure 22: Sample difficulty score distributions on HSOL before training and after each of five training epochs using BERT.

As shown in Figure 22, the HSOL dataset is highly imbalanced both in terms of label distribution and initial difficulty scores, with a large proportion of hard samples. After one training epoch, the number of easy samples increases slightly, indicating some initial learning progress. However, even after training is completed, a substantial number of difficult samples remain, suggesting that the model struggles to learn from a significant portion of the data.

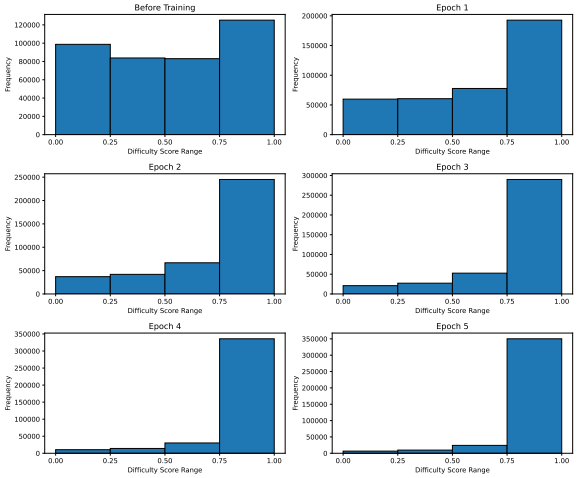

Figure 23: Sample difficulty score distributions on XNLI before training and after each of five training epochs using BERT.

As shown in Figure 23, the XNLI dataset exhibits a relatively balanced initial distribution of difficulty scores. Throughout training, both easy and difficult samples gradually increase or decrease in number in a stable manner, indicating consistent

learning dynamics. This stable progression may be attributed to the large size and diversity of the dataset, which provides sufficient training signals across difficulty levels.