Self-supervised Data Augmentation for Text Classification in Low-Data Settings

Deyu Ding[1], **Mengying Wang**[2], **Andreas Spitz**[2]
[1]Southern University of Science and Technology, Shenzhen, China
[2]University of Konstanz, Konstanz, Germany
deyu.ding@outlook.com, mengying.wang@uni.kn, andreas.spitz@uni.kn

Abstract

Due to data sparsity and high annotation cost, data augmentation has established itself as an effective tool for boosting model performance on supervised NLP tasks. Where task-agnostic augmentation methods tend to act as simple regularizers for the data, task-aware methods also leverage labels for the generation of data that are most suitable for downstream tasks. While prior work has investigated generation and sampling strategies individually, the potential of a self-supervised approach that leverages multiple pre-trained models in generation and sampling remains underexplored. To address this issue, we present an ensemble-based framework of language models that proposes augmentation candidates and internally reviews their suitability for low-resource text classification tasks. We evaluate our model on six classification benchmarks and find that it consistently outperforms state-of-the-art data augmentation baselines in classification accuracy by an average of 0.97 points in low-data scenarios.

Keywords: Data Augmentation, Pre-trained Language Models, Ensemble-based Self-supervision

1. Introduction

Data augmentation (DA), the process of inferring novel examples from a limited set of data, is a commonly used technique in machine learning for NLP, where it is particularly useful in low-data settings that impose constraints on the availability of labeled data. Compared to DA in computer vision or signal processing, the generation of augmented data in NLP is more challenging due to the discrete nature of human language that does not allow for simple interpolation or transformation techniques.

It is commonly acknowledged that data augmentation works similarly to regularizers (Zhang et al., 2017; Hernández-García and König, 2018; Dao et al., 2019). However, there are few conclusive answers regarding the fundamental mechanisms that make data augmentation effective, similar to the link between regularization and model generalization (Zhang et al., 2021). For task-aware DA, empirical research indicates that a crucial advantage is the exploitation of domain knowledge (Hernández-García and König, 2018), which is further supported by the finding that using language models in the generation of augmented data almost always outperforms rule-based techniques (Rashid and Amirkhani, 2021). Furthermore, when using pre-trained transformer models for classification, task-agnostic DA techniques have been shown to provide sparse and inconsistent improvements as those models may benefit more from task-aware DA techniques providing linguistic patterns that are relevant to the task but not seen during pre-training (Longpre et al., 2020). Based on these observations, we expect that the use of multiple pre-trained language models (PLMs) for data generation can leverage diversity in their pre-training data to increase the odds of creating augmented data that is beneficial to the performance of the downstream model. In particular, we conjecture that such an augmentation technique is especially useful in low-data scenarios, where the availability of original data is constrained.

Contributions. We propose an ensemble-based method for task-aware data augmentation that leverages a diverse selection of PLMs to generate candidates, which are reviewed and selected based on their suitability for the downstream task in a self-supervised setup. We demonstrate our method's performance against state-of-the-art baselines on classification benchmarks, and show that we reach competitive performance with as little as 5% of the original training data, indicating suitability for scenarios with few available data.

2. Related Work

Data augmentation in NLP can coarsely be grouped into three categories (Feng et al., 2021), of which we consider representatives in our framework: rule-based techniques, example interpolation techniques, and model-based techniques. We discuss these techniques in general, before touching on the most closely related work in sample-efficient data augmentation and data augmentation with PLMs.

2.1. Data Augmentation Background

2.1.1. Rule-based Techniques

In a relatively early work, Şahin and Steedman (2018) propose dependency tree morphing, lever-

Proceedings of the Fifteenth Language Resources and Evaluation Conference (LREC 2026), pages 10046–10056
11–16 May 2026.

aging knowledge of the syntactic structure to augment examples. Wei and Zou (2019) propose easy data augmentation (EDA), a set of token-level random perturbation operations including synonym replacement, random insertion, random swap, and random deletion. Among these, only synonym replacement injects prior knowledge, yet random deletion may inversely cause information loss, while random insertion and random swap may introduce distorted information. To eliminate these issues, Karimi et al. (2021) subsequently propose an easier data augmentation (AEDA), which includes only random insertion of punctuation marks into original text, outperforming the EDA in all five datasets they experiment.

2.1.2. Interpolation Techniques

Zhang et al. (2018), who interpolate the inputs and labels of two real data examples to create augmented data, find that interpolating only between inputs with matching labels does not improve the performance. Chen et al. (2020) propose MixText, which further explores interpolation between labeled and unlabeled data in the higher hidden layers of pre-trained transformer models by means of a modified consistency training. A shortcoming of these interpolation techniques is that they only consider the linear relationship between original examples and lack diversity in the augmented data. To tackle these two problems, Xie et al. (2022) propose Global Mixup, which separates augmented data generation and label determination. In line with this thinking, Zheng et al. (2023) differentiate the original data based on their difficulty and train the model in an evolutionary approach.

2.1.3. Model-based Techniques

Sennrich et al. (2016) propose a backtranslation method, which translates the original sentence into different intermediate languages and then back to obtain augmented data. Kobayashi (2018) modify a language model to contextualize the label information for the generation of augmented data. Anaby-Tavor et al. (2020) adapt GPT-2 (Radford et al., 2019) into a conditional augmented data generator and obtain substantial improvements in scenarios with scarce data. Rather than fine-tuning a single GPT-2 model to generate augmented data with labels of all kinds, Bayer et al. (2023) take a more fine-grained approach by fine-tuning a GPT-2 for each label group and separately contextualize long and short instances of text.

2.2. Sample-efficient Data Augmentation

In a review of augmentation techniques, Gontijo-Lopes et al. (2021) demonstrate empirically that good DA techniques tend to provide a balance between affinity and diversity. As the quality of augmented data tends to be unstable, inferior augmented data may be generated that is redundant or even harmful. To tackle this problem, Kamalloo et al. (2021) propose a sample-efficient DA strategy tailored for knowledge distillation, which they further generalize into a universal sample-efficient framework (Kamalloo et al., 2022). In contrast, Wang et al. (2023) directly embed a prompt design.

Although it embodies the same core idea as the above works, our data augmentation approach differs by (1) the integration of multiple distinct generation techniques, (2) the use of multiple neural models in the example selection step, and (3) the addition of a syntax correction module.

2.3. Data Augmentation with PLMs

For task-agnostic data augmentation of natural language, pre-trained language models are an obvious choice since their masked language modeling or generative capability can be used to generate or alter text. For conditional data augmentation that takes into account the label information, Wu et al. (2019) fine-tune a contextualized BERT model on downstream classification tasks by substituting segment embeddings for label embeddings. In contrast, Kumar et al. (2020), incorporate label information by prepending or appending it to the sentences. In a more aggressive approach, Anaby-Tavor et al. (2020) exploit the decoder architecture of a transformer model to generate entirely new sentences instead of making localized changes. This approach is further explored by Bayer et al. (2023) in a more fine-grained way, who fine-tune GPT-2 for each group of data labels. More robustly, Kumar et al. (2020) compare the performance of DA using different pre-trained models, including autoencoder models, autoregressive models and sequence-to-sequence models. Among all 3 architectures, they find seq2seq model to perform best in catering to downstream tasks, reaching a balance between diversity and semantic fidelity.

In contrast to the above works, our approach does not rely on a single PLM to generate augmentation candidates, but leverages multiple PLMs.

3. Method

Our method is based on the intuition that an ensemble of (generative) language models can be used in conjunction with task-aware reviewer models to produce high-quality augmented data. Our approach consists of a set of **proposer models** that generate augmented candidate data, followed by a grammar correction step that checks the candidates for consistency, and a set of **reviewer models** that select

candidates based on their veracity for the downstream task for which data is augmented.

3.1. Proposer Models

The task of the proposer models is the generation of augmented text data. We consider three types of PLMs: autoencoders, autoregressors, and sequence-to-sequence models.

Autoencoders (AE). As models in this group group, we utilize BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), ALBERT (Lan et al., 2020) and DeBERTa (He et al., 2021), which differ in architectural details and their pre-training corpus, but share the same backbone of encoder transformers (Vaswani et al., 2017). To generate augmented sentences, we randomly mask 20% of tokens in the input and replace them with the most probable model output token for the mask.

Autoregressors (AR). We use GPT-2 (Radford et al., 2019) and Deepseek-R1 (DeepSeek-AI, 2025). To generate augmented sentences, we truncate the tailing 20% of tokens from the input data and let the model predict the continuation until it outputs end-of-sentence or twice as many output tokens are created as tailing tokens were deleted. When generating multiple candidates per input, we output the most probable generated sentences according to the beam search tree.

Sequence-to-sequence Models (Seq2seq). As a seq2seq model, we utilize BART (Lewis et al., 2020) and follow the same design as Kumar et al. (2020) by masking a contiguous span of 20% of the words in the input sentence and using the model to fill in the blanks autoregressively.

Since the augmentation procedure may introduce syntactical errors, we conduct grammatical error correction for all candidate augmented data. Specifically, we use the model proposed by Rothe et al. (2021) to identify and correct errors before passing the data to the reviewers.

3.2. Reviewer Models

To select from the augmented candidates the ones that are most suitable for the downstream task, we employ an ensemble of reviewer models. Specifically, we use the pre-trained models BERT, RoBERTa, ALBERT, DeBERTa, and ELECTRA (Clark et al., 2020) and fine-tune them on the downstream task. For the fine-tuning of these models, we exclusively use the input data that is to be augmented (i.e., there is no data leakage from test to training data on the downstream evaluation), which we split into 90% training and 10% test data.

To determine the filtering criteria for candidate samples, several ensembling policies are possible, such as threshold-based pooling or majority voting. In preliminary experiments, we observed that requiring unanimous agreement among reviewers often led to overly strict filtering and reduced augmentation diversity. Therefore, we adopt a more permissive strategy: we discard candidates that are misclassified by all reviewer models and retain those for which at least one reviewer predicts the correct label. This approach balances label consistency with diversity, as different reviewer models may exhibit distinct decision boundaries and biases. The final augmented data is then sampled at random from the set of accepted candidates.

In Section 5.2, we conduct ablative tests to support these design decisions.

4. Evaluation

We evaluate the performance of our method on text classification tasks by comparing it to five baseline data augmentation techniques.

4.1. Evaluation Data

We conduct our experiments on six English classification datasets (see Table 1 for statistics).
SST, the Stanford Sentiment Treebank (Socher et al., 2013), is a dataset for sentiment classification on movie reviews. Two versions of this dataset are available: **SST-5** is annotated with five labels, and **SST-2** is annotated for binary classification.
SUBJ is labeled for binary classification and requires the classification of sentences as subjective or objective (Pang and Lee, 2004).
TREC (Li and Roth, 2002) is a question classification dataset, labeled for six classes (abbreviation, entity, description, human, location, and numeric).
YELP contains data from the business review website of the same name (Zhang et al., 2015), labeled with 5 classes. From the 650k training and 50k test data, we sample a small version (**YELP-S**) that is comparable in size to the other datasets, as well as a larger version (**YELP-L**).
YAHOO! ANSWERS (Zhang et al., 2015) is a topic classification dataset labeled with 10 main categories. Similar to YELP, we downsample the

dataset	classes	train	val	test
SST-2	2	6,920	872	1,829
SST-5	5	8,544	1,101	2,210
SUBJ	2	8,000	n/a	2,000
TREC	6	5,452	n/a	500
YELP-S	5	8,000	n/a	2,000
YAHOO-S	10	8,000	n/a	2,000
YELP-L	5	160,000	n/a	40,000
YAHOO-L	10	160,000	n/a	40,000

Table 1: Dataset statistics, including the number of classes, as well as the sizes of training, validation, and test sets. n/a: no default split was provided.

		dropout of (lyr)			number of		size of (lyr)			
	lr	att	cls	hid	att hd	hid lyr	hid	inm	optimizer	batch size
BERT	10^{-5}	0.1	0	0.1	12	12	768	3072	Adam	64
ALBERT	10^{-5}	0	0.1	0	12	12	768	3072	Adam	64
DeBERTa	10^{-5}	0.1	0	0.1	12	12	768	3072	Adam	64
DistilBERT	10^{-5}	0.1	0.2	0.1	12	6	768	3072	Adam	64
RoBERTa	10^{-5}	0.1	0	0.1	12	12	768	3072	Adam	64

Table 2: Hyperparameters of reviewer models. Listed are learning rate (lr); dropout of layers (lyr) for attention (att), classifier (cls), and hidden (hid); number of attention heads (att hd) and hidden layer (hid lyr); and size of the hidden (hid) and intermediate (inm) layers.

Dataset	no DA	BERT	RoBERTa	ALBERT	DeBERTa	ELECTRA
SST-2	90.73 ± 0.59	90.78 ± 0.62	91.12 ± 0.55	89.88 ± 0.64	**91.45 ± 0.50**	90.90 ± 0.58
SST-5	52.01 ± 1.48	51.84 ± 1.53	52.71 ± 1.35	51.20 ± 1.65	**53.12 ± 1.20**	52.43 ± 1.40
SUBJ	96.60 ± 0.56	96.48 ± 0.58	96.95 ± 0.51	95.85 ± 0.63	**97.28 ± 0.46**	96.75 ± 0.54
TREC	95.49 ± 0.72	95.38 ± 0.63	96.72 ± 0.57	95.60 ± 0.66	**96.89 ± 0.54**	96.30 ± 0.60
YELP-SUB	61.08 ± 0.67	61.32 ± 0.70	**62.17 ± 0.58**	60.63 ± 0.69	61.94 ± 0.53	61.40 ± 0.59
YAHOO-SUB	64.75 ± 0.64	64.58 ± 0.66	**65.83 ± 0.57**	64.17 ± 0.66	65.51 ± 0.52	65.12 ± 0.61

Table 3: Reviewer model performance across datasets. Numbers are accuracy (%) with standard deviation indicated by ±. Bold values represent the best performance for each dataset.

1.4M training and 60k test data points into a large (**YAHOO-L**) and a small (**YAHOO-S**) version.

4.2. Baselines

We compare our method against five recent additions to the data augmentation literature that provide working code repositories.

AEDA is a rule-based DA technique that randomly inserts a specified ratio of punctuation into the text (Karimi et al., 2021).

GLOBALMIXUP is an interpolation technique that utilizes clustering to mitigate the drawback of more traditional interpolation techniques that only consider linear relationships between two original examples (Xie et al., 2022).

BACKTRANS is a backtranslation approach (Sennrich et al., 2016), which involves translating text into an intermediate language and then back to the original language. We use German, French, Italian, and Chinese as intermediate languages.

GPT3MIX (Yoo et al., 2021) is a prompt-based approach that uses generative PLMs to regularize the linear interpolation (Zhang et al., 2018) between multiple different input sentences.

SSMBA (Ng et al., 2020) uses a manifold approach to generate synthetic data samples.

4.3. Experimental Setup

To classify the (augmented) data, we follow Munikar et al. (2019) in using a classification-tuned BERT-variant model (Devlin et al., 2019). As evaluation metric, we report accuracy. Standard deviations are derived from 10 repetitions of the experiments, using cross-validation. The hyperparameter settings and individual performances of all models used in our pipeline are listed as:

Downstream Classification Model Following Munikar et al. (2019), we fine-tune a BERT-variant model for classification, using a learning rate of 10^{-5}, a batch size of 64, and the Adam optimizer.

Grammar Correction To fix syntactic errors that may be introduced by the proposer models, we use the grammatical error correction model of Rothe et al. (2021), which is based on a T5-style encoder-decoder Transformer architecture for seq-to-seq correction. Since all datasets in our experiments are English, the model is applied exclusively to English text. The model performs full-sentence grammatical rewriting, with decoding via beam search with beam size 5, maximum sequence length 512, and early stopping.

Reviewer Models Reviewer models are trained for 10 epochs, after which we select the best-performing checkpoint of each model as the reviewer model. Table 2 lists the hyperparameters used for the reviewer models. In Table 3, we show the performance of the trained reviewer models.

To investigate the performance of data augmentation techniques specifically in scenarios with limited data availability, we evaluate the performance when 5, 25, 50, and 100% of the training data are used for augmentation and training while the remaining data are discarded. By default, we generate an amount of augmented data that is equivalent to 300% of the input data in all experiments. We vali-

dataset	%	no DA	ours	AEDA	GLOBALMIXUP	BACKTRANS	GPT3MIX	SSMBA
SST-2	5	84.11 ± 1.88	**87.43** ± 1.68	83.35 ± 2.24	85.55 ± 1.57	86.89 ± 1.87	86.53 ± 1.71	85.46 ± 1.88
	25	88.75 ± 0.81	**90.75** ± 0.71	88.98 ± 1.25	89.62 ± 0.73	90.44 ± 1.06	89.89 ± 0.73	89.68 ± 1.00
	50	89.51 ± 0.63	**91.17** ± 0.56	90.07 ± 0.84	90.39 ± 0.47	91.13 ± 0.69	90.97 ± 0.64	90.53 ± 0.68
	100	90.73 ± 0.59	**91.35** ± 0.50	91.01 ± 0.63	91.26 ± 0.45	91.34 ± 0.57	91.21 ± 0.45	91.21 ± 0.56
SST-5	5	44.34 ± 3.13	**49.73** ± 2.41	45.92 ± 3.35	45.66 ± 2.40	48.93 ± 2.58	48.63 ± 2.79	46.84 ± 2.98
	25	49.46 ± 2.06	**51.62** ± 1.39	50.44 ± 1.83	51.09 ± 1.41	50.90 ± 1.44	51.21 ± 1.43	50.81 ± 1.55
	50	50.93 ± 1.75	**52.99** ± 1.27	51.29 ± 1.70	51.91 ± 1.35	51.82 ± 1.35	51.49 ± 1.20	51.67 ± 1.46
	100	52.01 ± 1.48	**53.93** ± 0.89	52.28 ± 1.34	52.87 ± 1.13	52.01 ± 1.13	52.26 ± 1.09	52.39 ± 1.20
SUBJ	5	95.01 ± 1.78	**95.55** ± 1.61	94.43 ± 2.11	95.27 ± 1.62	95.05 ± 1.85	94.82 ± 1.91	94.92 ± 1.76
	25	95.80 ± 0.79	96.29 ± 0.73	95.37 ± 1.37	**96.43** ± 0.77	96.10 ± 0.84	95.72 ± 0.98	95.97 ± 1.00
	50	96.04 ± 0.61	96.78 ± 0.53	95.53 ± 1.18	**96.83** ± 0.69	96.44 ± 0.57	96.12 ± 0.52	96.57 ± 0.72
	100	96.60 ± 0.56	96.97 ± 0.48	95.80 ± 0.84	**97.05** ± 0.63	96.90 ± 0.56	96.45 ± 0.52	96.95 ± 0.58
TREC	5	91.00 ± 1.80	**94.23** ± 1.65	90.58 ± 2.06	92.76 ± 1.55	93.03 ± 1.78	93.36 ± 1.54	92.46 ± 1.78
	25	93.20 ± 0.83	**96.16** ± 0.72	92.95 ± 1.08	95.75 ± 0.64	95.14 ± 0.80	95.41 ± 1.05	94.61 ± 0.85
	50	94.16 ± 0.78	**97.04** ± 0.59	93.67 ± 0.99	96.90 ± 0.60	96.07 ± 0.79	96.21 ± 0.72	96.39 ± 0.69
	100	95.49 ± 0.72	97.26 ± 0.53	94.94 ± 0.83	**97.29** ± 0.53	96.49 ± 0.65	96.62 ± 0.71	96.99 ± 0.63
YELP-S	5	48.86 ± 2.35	**50.40** ± 1.95	48.35 ± 2.50	49.17 ± 2.12	48.99 ± 2.24	49.91 ± 2.18	48.84 ± 2.26
	25	58.84 ± 0.91	**60.04** ± 0.66	58.50 ± 1.17	59.35 ± 0.89	59.28 ± 1.03	59.40 ± 1.01	59.00 ± 1.01
	50	59.76 ± 0.75	61.40 ± 0.59	61.27 ± 1.06	61.12 ± 0.76	60.92 ± 0.72	**61.71** ± 0.73	61.10 ± 0.83
	100	61.08 ± 0.67	62.66 ± 0.47	62.41 ± 1.23	61.86 ± 0.53	61.74 ± 0.67	**62.77** ± 0.52	62.01 ± 0.74
YAHOO-S	5	58.67 ± 3.14	**63.17** ± 1.92	57.20 ± 3.25	60.26 ± 2.25	60.95 ± 2.41	61.85 ± 2.31	60.41 ± 2.55
	25	63.80 ± 0.92	**65.90** ± 0.69	63.03 ± 1.17	64.05 ± 0.84	65.11 ± 0.96	64.95 ± 0.84	64.68 ± 0.92
	50	64.48 ± 0.74	**66.66** ± 0.63	65.12 ± 0.79	65.89 ± 0.63	65.47 ± 0.79	65.63 ± 0.73	65.61 ± 0.72
	100	64.75 ± 0.64	**67.04** ± 0.55	65.25 ± 0.82	65.54 ± 0.65	65.79 ± 0.73	65.89 ± 0.63	65.63 ± 0.72
YELP-L	5	61.18 ± 0.57	**62.56** ± 0.58	62.32 ± 1.44	61.24 ± 0.60	61.84 ± 0.71	61.54 ± 0.58	61.13 ± 0.68
	25	65.23 ± 0.52	**66.33** ± 0.50	65.86 ± 1.48	65.72 ± 0.65	65.46 ± 0.69	65.36 ± 0.60	65.68 ± 0.77
	50	66.84 ± 0.49	67.25 ± 0.52	**67.45** ± 1.29	67.12 ± 0.68	67.19 ± 0.54	66.90 ± 0.52	67.09 ± 0.62
	100	68.05 ± 0.50	68.14 ± 0.53	68.17 ± 1.34	68.07 ± 0.56	67.88 ± 0.63	**68.45** ± 0.58	68.03 ± 0.60
YAHOO-L	5	64.87 ± 0.66	**66.77** ± 0.59	65.12 ± 0.85	65.34 ± 0.59	65.83 ± 0.69	65.76 ± 0.64	65.01 ± 0.68
	25	66.65 ± 0.54	**67.01** ± 0.55	66.75 ± 0.76	66.75 ± 0.53	66.47 ± 0.58	66.84 ± 0.52	66.65 ± 0.60
	50	67.21 ± 0.52	**67.45** ± 0.50	67.40 ± 0.73	67.12 ± 0.49	66.98 ± 0.57	67.17 ± 0.46	67.17 ± 0.55
	100	67.69 ± 0.58	67.53 ± 0.46	**67.71** ± 0.79	67.25 ± 0.47	67.13 ± 0.57	67.28 ± 0.50	67.37 ± 0.55

Table 4: BERT classification accuracy (± standard deviation) for data augmentation techniques when 5, 25, 50, and 100% of the training data are used as input for data augmentation. The best performing method per row is highlighted.

date the impact of the amount of augmented data independently in an ablation test in Section 5.2.

5. Results

This section reports our experimental findings. We first present the main quantitative results in Sec 5.1 and then conduct ablation studies in Sec 5.2 to analyze the influence of individual components.

5.1. Experimental Results

We show an aggregated view of our findings in Figure 1 and the full experimental results in Table 4.

On the six small datasets, the performance trends are comparable: our method consistently outperforms all baselines in the low-data scenarios where 5% or 25% of the data are used for augmentation and training. In the 5% scenario, we outperform the closest baseline by 0.97 accuracy points on average. For the larger percentages of data used, performance differences between the methods diminish and all methods provide comparable performances. Notably, on all datasets except YELP, using our method to augment just 25% of the training data already provides a comparable performance to training a classifier on 100% of the original data, indicating that it can be a valuable tool in situations where data annotation is costly.

On the two large datasets, our method still provides top performance in the low-data scenarios, but the performances of all methods are overall close. Finally, performance differences between methods become virtually indistinguishable in the 100% scenario on large data sets, indicating that data augmentation is unnecessary in these cases.

Overall, these findings provide empirical evidence that our methods achieves at balance between affinity and diversity in creating augmented samples (Gontijo-Lopes et al., 2021), at least when few data are available.

5.2. Ablation Study

To assess the impact of design decisions on our model and the impact of the amount of generated augmented data on its performance, we conducted the ablation experiments described in the following.

5.2.1. Grammar Correction (GC)

When comparing the performance of our approach with and without the grammar correction step, we find that the downstream accuracy remains largely

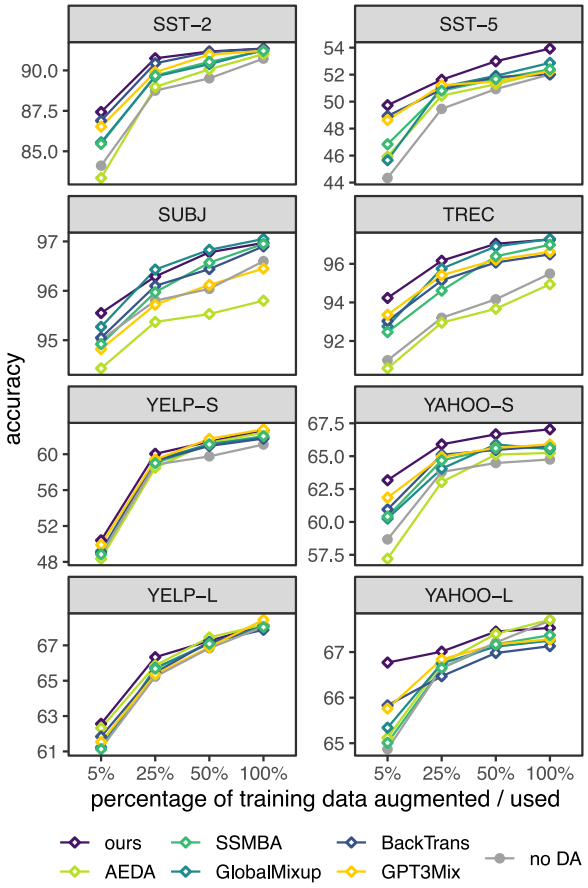

Figure 1: Downstream classification accuracy for all tested data augmentation methods and the augmentation-less baseline (no DA).

unaffected (see column **w/o GC** in Table 5). However, Table 6 shows the number of epochs that are required for convergence of the downstream classifier during evaluation, revealing that incorporating grammar correction reduces training time and enhances the stability of the training process. Based on this observation and the low compute that is required by grammar checking, we include this step in our final model.

5.2.2. Reviewing Module

Second, we evaluate the effectiveness of our review mechanism. As shown in column **w/o review** in Table 5), removing this component would substantially degrade performance, especially in low-data scenarios.

5.2.3. Training on Perplexing Candidates

Third, we test the hypothesis that exclusively using perplexing candidates (i.e., generated candidates for which at least one reviewer model makes an incorrect prediction while at least one reviewer model makes a correct prediction) for data augmentation could improve downstream performance by better sampling around the decision boundary. As shown in the column **perplexing** in Table 5, training solely on this subset can sometimes outperform our approach of also retaining augmented data for which all reviewer models made correct predictions. However, the performance gain is marginal and inconsistent, suggesting that an exclusive focus on perplexing examples would offer no added benefits.

5.2.4. Training on Easy Candidates

Conversely, we also investigate the effect on exclusively training on augmented data for which at least three reviewer models provide a correct prediction. The results are shown in the column **easy** in Table 5, and indicate that (expectedly), this approach decreases downstream accuracy. When the training set is small, the effect of filtering low-quality data is limited. However, as the number of training data increases, the accuracy decrease worsens.

Overall, these ablation tests indicate that a balanced augmented data composition of perplexing and easy candidates provides the most robust downstream performance.

5.2.5. Augmentation Volume

To test the sensitivity of our approach to the amount of generated data, we conduct experiments in which we generate 100%, 200%, and 300% of the original data volume in augmented data. As shown in Table 7, more data consistently results in an increased performance, which is why we use 300% augmented data in all experiments. However, the gains in performance are relatively limited, so an argument can be made for using less augmented data in settings where compute is limited.

5.3. Runtime Analysis

For transparency, we also conducted a runtime comparison between our model and all baselines, which we show in Table 8. Unsurprisingly, using an ensemble of models for generating and reviewing candidates comes at a computational cost that exceeds single models and rule-based baselines. While the runtimes of our method are an order of magnitude above the next-best competitors GLOBALMIXUP and GPT3MIX (hours vs. minutes of runtime), this cost is not prohibitive. Since our method is designed specifically for settings in which few training data are available, the runtimes remain manageable even on modest hardware.

6. Summary and Conclusion

We propose a novel self-supervised data augmentation technique for low-data scenarios that consistently outperforms recent state-of-the-art baseline

Dataset	%	no DA	ours	w/o GC	w/o review	perplexing	easy
SST-2	5	84.11 ± 1.88	**87.43** ± 1.68	87.36 ± 1.69	86.35 ± 1.82	87.36 ± 1.71	87.21 ± 1.73
	25	88.75 ± 0.81	90.75 ± 0.71	**90.82** ± 0.68	89.55 ± 0.84	90.72 ± 0.74	90.72 ± 0.74
	50	89.51 ± 0.63	91.17 ± 0.56	**91.21** ± 0.58	90.66 ± 0.67	91.20 ± 0.58	91.20 ± 0.58
	100	90.73 ± 0.59	**91.35** ± 0.50	91.33 ± 0.50	91.06 ± 0.56	91.29 ± 0.52	91.29 ± 0.58
SST-5	5	44.34 ± 3.13	49.73 ± 2.41	**49.79** ± 2.47	47.29 ± 2.84	49.95 ± 2.35	49.72 ± 2.44
	25	49.46 ± 2.06	51.62 ± 1.39	**51.69** ± 1.44	50.91 ± 1.47	51.70 ± 1.37	50.72 ± 1.49
	50	50.93 ± 1.75	**52.99** ± 1.27	52.95 ± 1.31	51.63 ± 1.37	52.70 ± 1.31	51.47 ± 1.41
	100	52.01 ± 1.48	**53.93** ± 0.89	53.91 ± 0.93	52.35 ± 1.12	53.70 ± 0.93	52.20 ± 1.14
SUBJ	5	95.01 ± 1.78	95.55 ± 1.61	95.58 ± 1.65	94.97 ± 1.89	**95.62** ± 1.60	94.81 ± 1.87
	25	95.80 ± 0.79	96.29 ± 0.73	**96.33** ± 0.78	95.80 ± 0.91	96.32 ± 0.76	95.66 ± 0.94
	50	96.04 ± 0.61	96.78 ± 0.53	**96.80** ± 0.56	96.23 ± 0.73	96.73 ± 0.56	96.12 ± 0.75
	100	96.60 ± 0.56	**96.97** ± 0.48	96.93 ± 0.52	96.55 ± 0.61	96.92 ± 0.50	96.44 ± 0.62
TREC	5	91.00 ± 1.80	94.23 ± 1.65	94.20 ± 1.70	92.43 ± 1.68	**94.28** ± 1.63	92.26 ± 1.72
	25	93.20 ± 0.83	**96.16** ± 0.72	96.14 ± 0.74	94.31 ± 0.91	96.12 ± 0.75	94.17 ± 0.94
	50	94.16 ± 0.78	**97.04** ± 0.59	97.00 ± 0.60	95.21 ± 0.75	96.98 ± 0.60	95.06 ± 0.76
	100	95.49 ± 0.72	97.26 ± 0.53	97.28 ± 0.57	96.34 ± 0.66	**97.33** ± 0.56	96.20 ± 0.67
YELP-S	5	48.86 ± 2.35	50.40 ± 1.95	50.43 ± 2.00	48.85 ± 2.26	**50.49** ± 1.98	48.74 ± 2.27
	25	58.84 ± 0.91	60.04 ± 0.66	**60.10** ± 0.71	59.13 ± 1.03	60.00 ± 0.68	58.99 ± 1.05
	50	59.76 ± 0.75	**61.40** ± 0.59	61.39 ± 0.64	61.26 ± 0.75	61.33 ± 0.63	61.09 ± 0.78
	100	61.08 ± 0.67	62.66 ± 0.47	**62.68** ± 0.52	62.20 ± 0.73	62.60 ± 0.50	62.08 ± 0.74
YAHOO-S	5	58.67 ± 3.14	63.17 ± 1.89	63.15 ± 1.94	60.56 ± 2.57	**63.29** ± 1.86	62.61 ± 2.30
	25	63.80 ± 0.92	65.90 ± 0.69	65.94 ± 0.74	64.29 ± 0.88	**65.98** ± 0.70	64.14 ± 0.91
	50	64.48 ± 0.74	66.66 ± 0.63	**66.69** ± 0.68	65.53 ± 0.73	66.48 ± 0.67	65.44 ± 0.74
	100	64.75 ± 0.64	**67.04** ± 0.55	67.02 ± 0.60	65.62 ± 0.70	66.84 ± 0.60	65.46 ± 0.72
YELP-L	5	61.18 ± 0.57	62.56 ± 0.58	**62.59** ± 0.61	61.74 ± 0.66	62.51 ± 0.59	61.58 ± 0.69
	25	65.23 ± 0.52	66.33 ± 0.50	**66.36** ± 0.55	65.60 ± 0.70	66.31 ± 0.53	65.42 ± 0.72
	50	66.84 ± 0.49	67.25 ± 0.52	**67.28** ± 0.57	67.14 ± 0.54	67.19 ± 0.55	67.03 ± 0.58
	100	68.05 ± 0.50	68.14 ± 0.53	68.10 ± 0.58	**68.15** ± 0.59	68.08 ± 0.54	68.01 ± 0.61
YAHOO-L	5	64.87 ± 0.66	**66.77** ± 0.59	66.75 ± 0.63	65.51 ± 0.67	66.73 ± 0.58	65.42 ± 0.48
	25	66.65 ± 0.54	67.01 ± 0.55	**67.03** ± 0.60	66.70 ± 0.59	66.98 ± 0.53	66.59 ± 0.61
	50	67.21 ± 0.52	**67.45** ± 0.50	67.43 ± 0.55	67.17 ± 0.54	67.35 ± 0.54	67.02 ± 0.56
	100	**67.69** ± 0.58	67.53 ± 0.46	67.51 ± 0.51	67.35 ± 0.56	67.30 ± 0.49	67.21 ± 0.57

Table 5: Ablation experiments of our method based on BERT classification accuracy (± standard deviation) with 5, 25, 50, and 100% of training data.

dataset	no DA	ours	ours (w/o GC)
SST-2	3.6	5.1	5.7
SST-5	4.7	6.7	7.5
SUBJ	2.7	4.5	5.1
TREC	3.8	5.5	6.6

Table 6: Number of epochs needed for convergence using different DA strategies.

models by leveraging the capabilities of multiple pre-trained language models. While this performance comes at a steep computational cost, we find minimal performance gains by means of data augmentation across all tested models in data-rich scenarios where this would be reason for concern. Instead, we propose to use our technique in cases where few data are available and where the cost of an extensive manual annotation would outweigh the computational overhead.

7. Limitations

In addition to the runtime requirements, which directly entail hardware requirements and an increased carbon footprint, further experiments should be considered. While we performed extensive experiments with regard to data sets and baseline models, future work should investigate the performance of individual proposer models, ideally to identify correlations between specific proposer models and downstream tasks. Such results would enable a deliberate selection of (and reduction in) used models in the ensemble. Similar experiments could be considered for reviewer models.

Computational Cost and Scalability. Our augmentation pipeline uses multiple proposer and reviewer models, which leads to higher computational cost compared to simpler baselines. However, the augmentation step is performed offline and only once per dataset – meaning that the augmented

Dataset	%	no DA	ours(300%)	ours(200%)	ours(100%)
SST-2	5	84.11 ± 1.88	87.43 ± 1.68	87.47 ± 1.76	**87.53** ± 1.80
	25	88.75 ± 0.81	**90.75** ± 0.71	90.34 ± 0.77	90.02 ± 0.79
	50	89.51 ± 0.63	**91.17** ± 0.56	91.04 ± 0.59	90.94 ± 0.61
	100	90.73 ± 0.59	**91.35** ± 0.50	91.20 ± 0.51	91.11 ± 0.53
SST-5	5	44.34 ± 3.13	**49.73** ± 2.41	49.12 ± 2.48	48.68 ± 2.52
	25	49.46 ± 2.06	**51.62** ± 1.39	51.27 ± 1.41	50.91 ± 1.43
	50	50.93 ± 1.75	**52.99** ± 1.27	52.60 ± 1.31	52.14 ± 1.35
	100	52.01 ± 1.48	**53.93** ± 0.89	53.63 ± 0.91	53.41 ± 1.00
SUBJ	5	95.01 ± 1.78	**95.55** ± 1.61	95.38 ± 1.66	95.18 ± 1.72
	25	95.80 ± 0.79	**96.29** ± 0.73	96.17 ± 0.78	96.01 ± 0.85
	50	96.04 ± 0.61	**96.78** ± 0.53	96.65 ± 0.55	96.54 ± 0.57
	100	96.60 ± 0.56	**96.97** ± 0.48	96.87 ± 0.49	96.78 ± 0.49
TREC	5	91.00 ± 1.80	**94.23** ± 1.65	93.72 ± 1.67	93.21 ± 1.69
	25	93.20 ± 0.83	**96.16** ± 0.72	95.63 ± 0.77	95.12 ± 0.81
	50	94.16 ± 0.78	**97.04** ± 0.59	96.68 ± 0.62	96.33 ± 0.66
	100	95.49 ± 0.72	**97.26** ± 0.53	97.12 ± 0.53	96.98 ± 0.54
YELP-S	5	48.86 ± 2.35	**50.40** ± 1.95	50.00 ± 2.08	49.61 ± 2.12
	25	58.84 ± 0.91	**60.04** ± 0.66	59.59 ± 0.68	59.17 ± 0.73
	50	59.76 ± 0.75	**61.40** ± 0.59	61.13 ± 0.60	60.85 ± 0.61
	100	61.08 ± 0.67	62.66 ± 0.47	62.71 ± 0.49	**62.77** ± 0.51
YAHOO-S	5	58.67 ± 3.14	**63.17** ± 1.89	62.50 ± 2.02	61.82 ± 2.12
	25	63.80 ± 0.92	**65.90** ± 0.69	65.45 ± 0.71	65.01 ± 0.72
	50	64.48 ± 0.74	**66.66** ± 0.63	66.27 ± 0.66	65.89 ± 0.68
	100	64.75 ± 0.64	**67.04** ± 0.55	66.95 ± 0.59	66.91 ± 0.61
YELP-L	5	61.18 ± 0.57	**62.56** ± 0.58	62.24 ± 0.60	61.92 ± 0.61
	25	65.23 ± 0.52	**66.33** ± 0.50	66.06 ± 0.51	65.79 ± 0.51
	50	66.84 ± 0.49	67.25 ± 0.52	67.29 ± 0.53	**67.32** ± 0.55
	100	68.05 ± 0.50	68.14 ± 0.53	68.20 ± 0.52	**68.25** ± 0.52
YAHOO-L	5	64.87 ± 0.66	**66.77** ± 0.59	66.65 ± 0.59	66.52 ± 0.60
	25	66.65 ± 0.54	**67.01** ± 0.55	66.97 ± 0.55	66.94 ± 0.56
	50	67.21 ± 0.52	67.45 ± 0.50	**67.48** ± 0.52	67.43 ± 0.53
	100	67.69 ± 0.58	67.53 ± 0.46	67.70 ± 0.49	**67.78** ± 0.50

Table 7: Parameter sensitivity experiment, showing the performance of downstream classification when 100%, 200% and 300% of the original data volume is generated as augmented data.

model	runtime (s)
AEDA	1.2×10^1
GLOBALMIXUP	1.2×10^3
BACKTRANS	8.6×10^1
GPT3MIX	1.8×10^2
SSMBA	7.2×10^3
ours	3.6×10^4

Table 8: Comparison of runtimes for generating augmented data on the SST-5 dataset (A40 GPU).

data can be reused for downstream training and evaluation without additional computational cost once it has been generated. Our work focuses on low-data settings, where offline data augmentation is commonly used. Furthermore, the framework is modular, consisting of components that can be adjusted depending on available resources. For example, fewer proposer models or a smaller set of reviewer models could be used to reduce runtime, at the cost of a small decrease in performance. Future work could further investigate ways to improve efficiency, such as model pruning, distillation, or selecting a subset of reviewer models in a more adaptive manner.

Architecture Design and Model Diversity. In our current design, the reviewer ensemble primarily consists of encoder-based classification models, which are used due to their strong and stable performance in classification tasks, as well as their relatively moderate computational cost. At the same time, the proposer component already incorporates diverse model types, including autoencoders, autoregressive models, and sequence-to-sequence architectures. Of course, it would be interesting to explore whether large generative LLMs could also serve as effective reviewer models, potentially providing additional diversity. Conversely, smaller

and more lightweight models may offer comparable performance with reduced computational overhead. Investigating these alternatives could further clarify the trade-offs between architectural diversity, efficiency, and augmentation quality.

Individual Proposer Contributions. While proposer diversity appears beneficial, we do not analyze the contribution of each proposer family separately in this work. Since all generated samples are filtered by grammatical correction and reviewer models, low-quality outputs are largely removed. A more detailed study of individual proposer contributions is left for future work.

Model Scope. Finally, we note that we were restricted by our available compute resources in the selection of models that we could evaluate. Future work should investigate whether our findings hold for larger, more recent generative models as proposers.

From an ethics point of view, we see no issue with the proposed approach itself. However, the use of pretrained LLMs naturally incurs the risk that biases (social or otherwise) in their pre-training data could leak into the augmented data.

8. AI Statement

AI tools were used as writing assistants in drafting internal versions of this manuscript. The final version was fully (re)written by the authors.

9. Bibliographical References

Ateret Anaby-Tavor, Boaz Carmeli, Esther Goldbraich, Amir Kantor, George Kour, Segev Shlomov, Naama Tepper, and Naama Zwerdling. 2020. Do not have enough data? deep learning to the rescue! In *The Thirty-Fourth AAAI Conference on Artificial Intelligence*.

Markus Bayer, Marc-André Kaufhold, Björn Buchhold, Marcel Keller, Jörg Dallmeyer, and Christian Reuter. 2023. Data augmentation in natural language processing: a novel text generation approach for long and short text classifiers. *Int. J. Mach. Learn. Cybern.*, 14(1):135–150.

Jiaao Chen, Zichao Yang, and Diyi Yang. 2020. MixText: Linguistically-informed interpolation of hidden space for semi-supervised text classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: pre-training text encoders as discriminators rather than generators. In *8th International Conference on Learning Representations, ICLR*.

Tri Dao, Albert Gu, Alexander Ratner, Virginia Smith, Chris De Sa, and Christopher Ré. 2019. A kernel theory of modern data augmentation. In *Proceedings of the 36th International Conference on Machine Learning, ICML*.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186.

Steven Y. Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. 2021. A survey of data augmentation approaches for NLP. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP*, pages 968–988.

Raphael Gontijo-Lopes, Sylvia Smullin, Ekin Dogus Cubuk, and Ethan Dyer. 2021. Tradeoffs in data augmentation: An empirical study. In *International Conference on Learning Representations*.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: decoding-enhanced bert with disentangled attention. In *9th International Conference on Learning Representations, ICLR*.

Alex Hernández-García and Peter König. 2018. Data augmentation instead of explicit regularization. *CoRR*, abs/1806.03852.

Ehsan Kamalloo, Mehdi Rezagholizadeh, and Ali Ghodsi. 2022. When chosen wisely, more data is what you need: A universal sample-efficient strategy for data augmentation. In *Findings of the Association for Computational Linguistics: ACL*.

Ehsan Kamalloo, Mehdi Rezagholizadeh, Peyman Passban, and Ali Ghodsi. 2021. Not far away, not so close: Sample efficient nearest neighbour data augmentation via minimax. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP*.

Akbar Karimi, Leonardo Rossi, and Andrea Prati. 2021. AEDA: an easier data augmentation technique for text classification. In *Findings of the Association for Computational Linguistics: EMNLP*.

Sosuke Kobayashi. 2018. Contextual augmentation: Data augmentation by words with paradigmatic relations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2020. Data augmentation using pre-trained transformer models. In *Proceedings of the 2nd Workshop on Life-long Learning for Spoken Language Systems*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *19th International Conference on Computational Linguistics, COLING*.

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. *CoRR*, abs/1907.11692.

Shayne Longpre, Yu Wang, and Chris DuBois. 2020. How effective is task-agnostic data augmentation for pretrained transformers? In *Findings of the Association for Computational Linguistics: EMNLP*, volume EMNLP 2020.

Manish Munikar, Sushil Shakya, and Aakash Shrestha. 2019. Fine-grained sentiment classification using BERT. *CoRR*, abs/1910.03474.

Nathan Ng, Kyunghyun Cho, and Marzyeh Ghassemi. 2020. SSMBA: self-supervised manifold based data augmentation for improving out-of-domain robustness. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP*.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Mohammad Amin Rashid and Hossein Amirkhani. 2021. The effect of using masked language models in random textual data augmentation. In *26th International Computer Conference, Computer Society of Iran, CSICC*.

Sascha Rothe, Jonathan Mallinson, Eric Malmi, Sebastian Krause, and Aliaksei Severyn. 2021. A simple recipe for multilingual grammatical error correction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP*.

Gözde Gül Şahin and Mark Steedman. 2018. Data augmentation via dependency tree morphing for low-resource languages. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*.

Richard Socher, John Bauer, Christopher D. Manning, and Andrew Y. Ng. 2013. Parsing with compositional vector grammars. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, ACL*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *30th Annual Conference on Advances in Neural Information Processing Systems*.

Congcong Wang, Gonzalo Fiz Pontiveros, Steven Derby, and Tri Kurniawan Wijaya. 2023. STA: self-controlled text augmentation for improving text classifications. *CoRR*, abs/2302.12784.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. 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 6382–6388.

Xing Wu, Shangwen Lv, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019. Conditional BERT contextual augmentation. In *19th International Conference on Computational Science ICCS*.

Xiangjin Xie, Yangning Li, Wang Chen, Kai Ouyang, Li Jiang, and Hai-Tao Zheng. 2022. Global mixup: Eliminating ambiguity with clustering. *CoRR*, abs/2206.02734.

Kang Min Yoo, Dongju Park, Jaewook Kang, Sang-Woo Lee, and Woo-Myoung Park. 2021. Gpt3mix: Leveraging large-scale language models for text augmentation. In *Findings of the Association for Computational Linguistics: EMNLP*.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2017. Understanding deep learning requires rethinking generalization. In *5th International Conference on Learning Representations, ICLR*.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2021. Understanding deep learning (still) requires rethinking generalization. *Commun. ACM*, 64(3):107–115.

Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. 2018. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *28th Annual Conference Advances in Neural Information Processing Systems*.

Haoqi Zheng, Qihuang Zhong, Liang Ding, Zhiliang Tian, Xin Niu, Changjian Wang, Dongsheng Li, and Dacheng Tao. 2023. Self-evolution learning for mixup: Enhance data augmentation on few-shot text classification tasks. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP*.