STAGE: Simple Text Data Augmentation by Graph Exploration

Ho-Seung Kim[1], **YongHoon Kang**[2], **Jee-Hyong Lee**[3†]
Sungkyunkwan University
Suwon-si, Republic of Korea
{tree901024[1], artfjqm2[2], john[3†]}@skku.edu

Abstract

Pre-trained language models (PLMs) are widely used for various tasks, but fine-tuning them requires sufficient data. Data augmentation approaches have been proposed as alternatives, but they vary in complexity, cost, and performance. To address these challenges, we propose **STAGE** (Simple Text Data Augmentation by Graph Exploration), a highly effective method for data augmentation. **STAGE** utilizes simple modification operations such as insertion, deletion, replacement, and swap. However, what distinguishes **STAGE** lies in the selection of optimal words for each modification. This is achieved by leveraging a word-relation graph called the co-graph. The co-graph takes into account both word frequency and co-occurrence, providing valuable information for operand selection. To assess the performance of **STAGE**, we conduct evaluations using seven representative datasets and three different PLMs. Our results demonstrate the effectiveness of **STAGE** across diverse data domains, varying data sizes, and different PLMs. Also, **STAGE** demonstrates superior performance when compared to previous methods that use simple modification operations or large language models like GPT3.

Keywords: data augmentation, text modification, text classification

1. Introduction

Text classification is a basic task that can solve many problems in NLP. There are various types of text classification, such as sentiment analysis, subjectivity classification, question classification, and pro-con classification. Recently, pre-trained language models (PLMs) (Clark et al., 2020; He et al., 2020; Lan et al., 2019; Wu et al., 2020a) based on the transformer have been widely used to solve the text classification problems (Dai and Le, 2015; Matthew, 2018; Radford et al., 2018; Howard and Ruder, 2018). PLMs show superior performance than any other existing models just by fine-tuning with the dataset for the target task.

A common challenge with PLM is that it needs enough data to perform downstream tasks. Data augmentation (DA) is one of the alternatives to solve the problem (Chen et al., 2023; Kumar et al., 2020; Şahin, 2022). Text DA approaches can be divided into two types: text generation and text modification. The text generation is an approach to generate sentences by deep learning (DL) models (Dong et al., 2019; Zhang et al., 2019). An easy example is back-translation (Edunov et al., 2018a), where a model trained on various languages can generate many sentences with similar meanings to the original sentence. However, it requires a lot of computation to generate sentences, and the diversity of generated sentences may be relatively limited.

On the other hand, text modification is a cost-effective method to generate a diverse set of sentences based on modification operations. However, in many previous approaches, the focus was primarily on developing new modification operations, with little consideration given to the selection of operands. Most of them randomly chose operands for the operations. While these approaches were capable of generating a variety of sentences, their performance suffered due to the random selection of operands, resulting in relatively low performance (Niu and Bansal, 2018). It is crucial to not only consider the modification operations themselves but also the selection of operands (words) to which the operations will be applied.

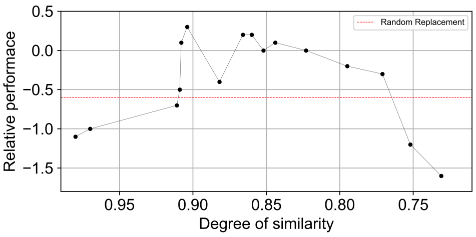

Figure 1: Augmentation performance by replace. It shows the relative performance and the similarity between the replaced word and its replacement. If replacement words are carefully chosen, the performance can be more improved.

Figure 1 shows the significance of operand selection. We conduct augmentation using the replace operation for the text classification task with SST-2 (Socher et al., 2013). We generate new sentences by replacing a word. The replacement word is chosen based on the similarity by Word2Vec. Figure

[†] Corresponding author

LREC-COLING 2024, pages 15238–15256
20-25 May, 2024.

1 shows the relative performance with respect to the similarity of the replacement word to the original word. It is very interesting that the performance varies much depending on which word is chosen for replacement despite utilizing the same operation. When a word is replaced by a word with a similarity of about 0.9, the augmentation performances are the best. We can observe similar patterns with other modification operations. We present more experimental observations in Figures 3 and 4 in the Appendix A. This shows the operand selection is also crucial.

When selecting operands, it is important to consider various factors such as importance, semantics, relations, and frequency. Since each modification operation injects a different type of noise, we need to choose operands considering such characteristics. For instance, in the delete operation, the importance of words can be a significant consideration, while in the replace operation, the semantic relationship of words needs to be taken into account.

To select suitable operands for each modification operation, we generate a word-relation graph from a corpus, considering term frequency and word co-occurrence. Based on this, we develop various methods to recommend operands suitable for each operation. We evaluate those and select the best ones. Finally, we propose our DA technique called **STAGE** (Simple Text Data Augmentation by Graph Exploration) by combining the chosen methods. Our **STAGE** is a a simple and novel text modification DA, and enables the selection of diverse operands by utilizing word importance, semantics, relations, and frequency information obtained from the word-relation graph.

To evaluate the performance, we apply **STAGE** to seven text classification tasks and three PLMs on various amount of train data. Compared to the previous methods, **STAGE** is generally effective. Especially, it shows an improvement up to 47% when the data is too small. Furthermore, **STAGE** offers the advantage of low computational cost and the ability to augment input data, making it suitable for low-resource settings and easily applicable in various environments.

2. Related Works

There are two main categories of Text DA approaches: text generation and text modification. Text modification is a DA approach employing modification operations such as deletion, insertion, etc. Some were token-level which modified individual tokens by applying simple operations to randomly chosen words (Karimi et al., 2021; Shou et al., 2022; Kolomiyets et al., 2011; Kobayashi, 2018; Luo et al., 2021; Şahin and Steedman, 2019; Zhong et al., 2020). One popular method is EDA (Wei and Zou, 2019). EDA generated sentences by applying simple operations on randomly chosen words. These methods are simple and effective, but the improvements were limited due to the randomness in selecting operands. As mentioned in Section 1, operands play a crucial role, yet they are often given little consideration in the previous methods.

Some methods focused on modifying a span of the sentence, rather than tokens (Wu et al., 2020b; Miao et al., 2020; Yu et al., 2019; Yoon et al., 2021). They chose an important span of sentences with additional information such as saliency, and modified the span. However, sentences were modified too much by span modification, resulting in performance degradation. There are also text modification methods that utilize graphs. AMR-DA (Shou et al., 2022) augmented text data by AMR parser and AMR graph. This method generated better samples than EDA, but the process was extremely complex and expensive.

Unlike text modification, the text generation aims to generate natural sentences (Lee et al., 2021; Kim and Kang, 2022; Zhao et al., 2017; Xie et al., 2020; Wu et al., 2019; Malandrakis et al., 2019). A representative method is back-translation (Edunov et al., 2018a; Sennrich et al., 2015; Edunov et al., 2018b), which generates sentences by re-translating sentences that had been translated into other languages using a pre-trained translation model. It requires a lot of computation because it has to go through two inference processes for DA. There are approaches which generate sentence representations, such as Text Smoothing (Wu et al., 2022), WordMix (Guo et al., 2019), and TMix (Chen et al., 2020). Since they generated new sentence representations instead of new sentences, they are not model-transparent, and hard to apply to fine-tuning of pre-trained LMs. Recently, there has been a significant amount of research on augmentation methods that utilize GPT-3 to specify templates and generate sentences accordingly like GPT3Mix (Yoo et al., 2021) and AugGPT (Dai et al., 2023).

3. Co-graph

Our method, **STAGE**, is a text modification approach using four simple operations: 'Delete', 'Replace', 'Insert', and 'Swap'. In order to choose their operands, we build a graph of word co-occurrence, abbreviated as co-graph, from text data, which can model the characteristics of words and word to word relations. Most existing modification methods depended on mainly randomness, but our method utilizes additional information about importance, semantic, relation, term frequency obtained from the co-graph.

3.1. Co-graph generation

We build the co-graph based on term frequency and co-occurrence to model the collocation information. All sentences in the training dataset are tokenized, and the unique tokens are used as nodes. We count if two words appear together within a window size of w in a sentence. We connect them if the count is higher than a threshold, τ. There can be some isolated nodes in the co-graph without any edges. We refer to the nodes that are not isolated in the co-graph as edge-nodes or edge-words. The nodes connected to a node in the co-graph is referred to as neighbor-nodes or neighbor-words of the corresponding node.

We also build sentence graphs (sen-graphs) for single sentences. A sen-graph is a sub-graph of the co-graph. The nodes of a sen-graph are the tokens in the corresponding sentence, and two nodes in the sen-graph is connected if they are connected in the co-graph.

3.2. Understanding Co-graph

Co-graph may represent two kinds of information. We can obtain strongly associated words in semantic and grammatical viewpoints. If an edge exists between two nodes in the graph, two words may be highly correlated because they appear together more than τ times. They may have a lexical and grammatical collocation. For example, the word ‘movie’ is connected with words like ‘star’, ‘horror’, ‘making’, ‘seen’, ‘great’, ‘kind’, and so on in movie review datasets. These lexical and grammatical collocations have important roles in fluency and idiomatic language production.

We also obtain the importance of words by the number of edges. If a word have few edges, such as isolated words, they may not be important because they have very weak association with other words. If an edge-word have a large number of edges, they may be also less important. Most of such words can be stop words because stop words are frequently used with various words (Silva and Ribeiro, 2003). Edge-words have with a modest number of edges can be considered important because they have strong associations with specific words.

We design our method based on these two aspects of co-graph. Since we can obtain word to word relations and importance of each word, we can choose better words for text modification based on these features. For example, in replace, we choose a word to be replaced based on the importance, and choose a word to replace among the collocated words. Through this, we can modify sentences while adjusting the degree of similarity.

4. Proposed Operations

The operations used in **STAGE** are ‘Delete’, ‘Replace’, ‘Insert’, and ‘Swap’, which are frequently used in previous text modification. As mentions in Introduction, we need to carefully select operands for text modification. We suggest a various way to choose operands based on the co-graph for each operation, because we can obtain various information on words from the co-graph has information on relations between words. Based on this, we propose sophisticate modification methods for each operations by combining operations and how to choose operands. We evaluate all methods and choose the optimal method for each operation. Finally, we combine them to propose **STAGE**. We describe proposed methods for each operation in the section, and the evaluation and combination of them in the next section. We present the summary of all the proposed modification operations in Table 12 in the Appendix B.

4.1. Delete operation

We propose four methods for delete: *D-RE*, *D-RI*, *D-ME*, and *D-LE*. For delete operations, we focus on edge-words which are frequently used together with another words. Thus, deleting such a word can effectively disrupt some frequent patterns, and create a significant change in the sentence. We also aim to select important or unimportant edge-words based on the number of edges.

D-RE deletes a random edge-word, and *D-RI* deletes an isolated word. They are simple delete methods and will be used as baselines. *D-ME* deletes the edge-word with the most edges in the sentence. A word with many edges means that it is frequently used with many other words regardless of patterns such as stop-words. *D-LE* deletes the edge-word in the sentence with the least edges. It may be a token frequently used in some special patterns, such as idioms. We aim to delete unimportant and important edge-word through *D-ME* and *D-LE*.

4.2. Replace operation

We propose six methods for replace: *R-RC*, *R-RS*, *R-RDS*, *R-MC*, *R-MS*, and *R-MDS*. The replace operation selects two words: one from the sentence and the other from the corpus. In previous approaches, a word was usually replaced with one of synonyms (Wu et al., 2020b; Wei and Zou, 2019; Shou et al., 2022). However, it is difficult to see this method as effective on performance, and there are cases where it degrades (Zhang et al., 2015). We choose two words based on the co-graph because the co-graph represents word-word relations. If two words are connected, they may have a lexical

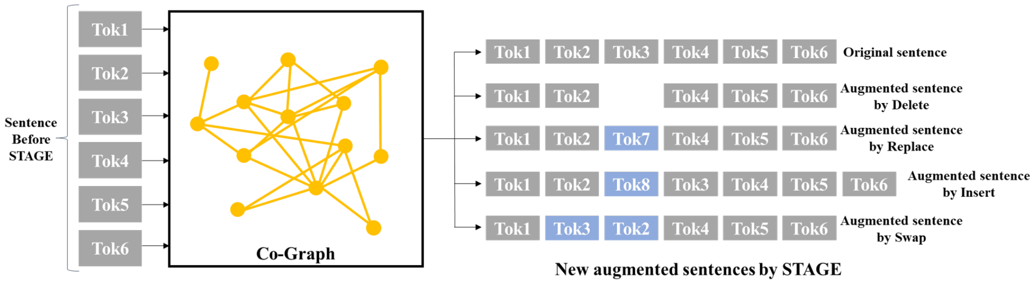
Sentence Before STAGE
Tok1
Tok2
Tok3
Tok4
Tok5
Tok6
Co-Graph
Tok1 Tok2 Tok3 Tok4 Tok5 Tok6 Original sentence
Tok1 Tok2 Tok4 Tok5 Tok6 Augmented sentence by Delete
Tok1 Tok2 Tok7 Tok4 Tok5 Tok6 Augmented sentence by Replace
Tok1 Tok2 Tok8 Tok3 Tok4 Tok5 Tok6 Augmented sentence by Insert
Tok1 Tok3 Tok2 Tok4 Tok5 Tok6 Augmented sentence by Swap
New augmented sentences by STAGE

Figure 2: Overview of **STAGE**. A sentence is augmented by a combination of these operations basic operations such as 'Delete', 'Replace', 'insert' and 'Swap'. We can obtain 5 samples for one input sentence.

and grammatical collocations. We can generate sentences while adjusting the degree of similarity using the co-graph.

R-RC, *R-RS* and *R-RDS* randomly select one of edge-words in the sentence, and replaces it with one of its neighbor-words in the co-graph. *R-RC* replaces the chosen edge-word with a random neighbor-word. *R-RS* and *R-RDS* replace the chosen edge-word with the most similar neighbor-word and with the most dissimilar neighbor-word, respectively. *R-RS* may generate a sentence with a similar meaning because it chooses the most similar one, and *R-RDS* may generate a sentence with a different meaning because it chooses the most dissimilar one.

In *R-MC*, *R-MS*, and *R-MDS*, we try to replace the most important words in the sentence. To choose important words, we use the sen-graph. We choose the word with the most edge in the sen-graph. We can say that such words are important words in the sentence because those words frequently appear together with most of other words in the sentence. *R-MC*, *R-MS*, and *R-MDS* replace the chosen word with a random, the most similar, and the most dissimilar word among neighbor-words in the co-graph, respectively.

4.3. Insert operation

We propose four methods for insertion: *I-EE*, *I-REN*, *I-MEN*, *I-LEN*. The insert operation selects a word from a corpus and insert it into the sentence at certain position. Previous approaches usually inserted a random word into a random position. A random word may change sentences too much in grammatically and semantically, and is limited to generate various sentences. In order to generate various sentences close to the original sentence, we use co-graph to select words.

I-EE randomly selects an edge-word in the sentence and one of its neighbor-words. It inserts the neighbor-word next to the edge-word in the sentence. While *I-EE* first select a position, and then select a word to be inserted considering the position, the others select a word and insert it at a random position. *I-REN*, *I-MEN* and *I-LEN* select a word from the co-graph words connected to the edge-words in the sentence. *I-REN* selects one at random, *I-MEN* selects the word connected to the most edge-words in the sen-graph, and *I-LEN* selects the word connected to the least edge-words in the sen-graph. *I-MEN* selects operand that are highly related to the entire sentence, and *I-LEN* selects lowly related one.

4.4. Swap operation

We propose three methods for swap: *S-RP*, *S-SP*, and *S-DSP*. If we swap two randomly chosen words, it may inject too much noise into the sentence. To decrease noise, we use two words connected in the sen-graph. If there is an edge in the graph, words often appear close to each other within a window size of w, frequently. If we swap such a pair, we can minimize noise. *S-RP* randomly selects a word pair. *S-SP* and *S-DSP* choose a pair based on word similarity. *S-SP* chooses the pair with the maximum similarity, and *S-DSP* chooses the pair with the minimum similarity. We evaluate the word similarity based on vector similarity. *S-DSP* has major impact, and *S-SP* minor impact on the meaning of sentence. *S-RP* may have an average of them.

5. STAGE: Combination of Operations

We propose **STAGE** (Simple Text Data Augmentation by Graph Exploration) by combining the proposed delete, replace, insert, and swap methods.

We have introduced several methods for each operation. However, utilizing a single operation alone has limitations in effectively augmenting text data. To address this, we aim to appropriately combine these four operations, leading to more effective DA. Figure 2 shows how text data can be augmented through various operations by **STAGE**. **STAGE** effectively enhances the diversity of text data by incorporating four operations.

To determine the optimal combination, we first evaluate the effectiveness of the proposed four

Method	How to augment data
D-Random	Delete the word randomly
D-TF-IDF high	Delete the highest TF-IDF score word
D-Word2Vec	Delete the most similar word with sentence
R-Random	Replace the word with other randomly
R-Word2Vec	Replace the word with synonym
I-Random	Insert the word randomly
I-Word2Vec	Insert the most similar word with sentence
S-Random	Swap the word pair randomly
S-Word2Vec	Swap the most similar word pair

Table 1: Simple data augmentation methods

Method	Training subset size ratio				Avg
	1%	10%	50%	100%	
BERT	55.7	86.7	89.7	91.2	80.8
D-Random	70.3	86.8	89.1	90.8	84.3
D-TF-IDF high	75.3	87.9	89.5	90.4	85.8
D-Word2Vec	72.7	87.5	89.3	90.9	85.1
D-RE (ours)	74.4	88.3	89.9	91.3	86.0
D-RI (ours)	70.4	87.2	89.2	90.9	84.4
D-ME (ours)	**75.4**	**88.4**	90.1	91.4	**86.3**
D-LE (ours)	74.1	88.2	**90.2**	**91.6**	86.0

Table 2: Evaluation of delete methods

Method	Training subset size ratio				Avg
	1%	10%	50%	100%	
BERT	55.7	86.7	89.7	91.2	80.8
R-Random	70.6	86.1	89.1	89.9	83.9
R-Word2Vec	70.6	87.1	89.5	90.5	84.4
R-RC (ours)	**79.6**	87.8	89.4	90.2	**86.8**
R-RS (ours)	77.5	87.6	89.6	90.8	86.4
R-RDS (ours)	74.3	86.8	89.6	91.1	85.5
R-MC (ours)	76.2	88.0	90.4	**91.7**	86.6
R-MS (ours)	76.0	**88.1**	89.9	91.6	86.4
R-MDS (ours)	72.5	88.0	**90.5**	91.6	85.7

Table 3: Evaluation of replace methods

Method	Training subset size ratio				Avg
	1%	10%	50%	100%	
BERT	55.7	86.7	89.7	91.2	80.8
I-Random	73.5	86.9	89.6	90.8	85.2
I-Word2Vec	71.7	87.3	90.0	91.0	85.1
I-EE (ours)	75.7	87.7	90.1	91.4	86.2
I-REN (ours)	77.2	88.4	90.2	91.6	86.9
I-MEN (ours)	**79.5**	**89.2**	90.2	**91.7**	**87.7**
I-LEN (ours)	78.1	88.5	**90.6**	91.6	87.2

Table 4: Evaluation of insert methods

deletion, six replacement, four insertion, and three swap operations. We evaluate the performance of each method using 4 subsets of the training dataset, each of which has 1%, 10%, 50% and 100% of the training dataset, respectively. The performance is measured in terms of accuracy. For the co-graph, we set the threshold τ to 10, and the parameter w to 2.

The evaluation is conducted on the SST-2 dataset (Socher et al., 2013). SST-2 exhibits various characteristics and is primarily considered a prominent dataset. It is used for assessing the effectiveness of sentiment analysis and text classifications. Although the dataset size is relatively small with 6,920 samples compared to other datasets, the high corpus-to-sentence ratio makes it a valuable resource for conducting experiments involving corpus handling.

With the evaluation results, we choose two most effective methods from each operation considering the performance for small corpora and large corpora respectively. We construct 16 STAGE candidates by combining them. We also evaluate 16 candidates with SST-2, and choose the best as **STAGE**.

5.1. Evaluation for each operation

We present the evaluation results of the proposed methods for each operation on the SST-2 dataset. We compare each methods with baseline methods which are based on randomness, TF-IDF, and Word2Vec. We obtain TF-IDF and Word2Vec values based on the training dataset. The baseline methods are shown in Table 1. In Tables 2, 3, 4, and 5, BERT is the performance of BERT without DA, and the others are the performance of BERT with DA by the corresponding methods. Due to space constraints, we provide a concise analysis. For a more detailed analysis and the whole experimental results, please refer to Tables 13, 14, 15, and 16, in the Appendix C.

5.1.1. Evaluation of delete methods

Table 2 shows the performance of delete operation methods. We can observe that all of our delete methods are superior to the baseline methods. The methods deleting edge-words, *D-RE*, *D-ME* and *D-LE* show higher performances than *D-RI* which deletes isolated words. It shows that deleting edge-words is more effective than isolated words. If we compare *D-ME* and *D-LE*, *D-ME* shows better performance and results differ depending on the subset size. We choose *D-ME* and *D-LE* for **STAGE**, which are the top 2 by the average performance.

5.1.2. Evaluation of replace methods

Table 3 shows the performance of replace operation methods. The proposed replace methods are also superior to the simple methods. *R-RC* and *R-MC* show higher overall performance than similarity-based methods, *R-RS*, *R-RDS*, *R-MS* and *R-MDS*. It can be noted that choose substitute words by the co-graph is more effective than by similarity. We choose *R-RC* and *R-MC* for **STAGE**, with high average performances.

Method	Training subset size ratio				Avg
	1%	10%	50%	100%	
BERT	55.7	86.7	89.7	91.2	80.8
S-Random	72.3	87.5	89.3	90.2	84.8
S-Word2Vec	72.5	87.2	90.0	90.5	85.1
S-RP (ours)	77.7	87.7	90.0	90.4	86.5
S-SP (ours)	**79.8**	87.6	89.8	90.3	**86.9**
S-DSP (ours)	72.3	**87.8**	**90.1**	**91.5**	85.4

Table 5: Evaluation of swap methods

5.1.3. Evaluation of insert methods

Table 4 shows the performance of insert operation methods. The proposed insert methods are also superior to the simple methods. *I-REN*, *I-MEN*, *I-LEN* choose a word to be inserted considering the correlation with all edge-words in the sentence. They have higher performance than *I-EE* which choose a word considering the word at the position to be inserted. We choose *I-MEN* and *I-LEN* for **STAGE**, with high average performances.

5.1.4. Evaluation of swap methods

Table 5 shows the performance of swap operation methods. The proposed methods show better performance than the others. *S-RP* shows that swapping pairs connected by a sen-graph is effective. Considering similarity *S-SP* and *S-DSP* are more effective depending on the size of the dataset. We choose *S-SP* and *S-DSP* considering the performance for small corpora and large corpora, respectively.

5.2. Optimal Combination for STAGE

As mentioned, we select the top 2 methods for each operation: *D-ME* and *D-LE* for delete, *R-RC* and *R-MC* for replace, *I-ME* and *I-LE* for insert, and *S-SP* and *S-DSP* for swap. We combine them to create 16 candidate STAGE combinations, and evaluate them with SST-2. Table 6 shows the performance of three combinations: the best (STAGE$_1$), the second best (STAGE$_2$), and the worst (STAGE$_{16}$) combinations. The experimental results of the all 16 combinations can be found can be found in Table 17 in the Appendix D.

Finally, the combination of D-ME, R-MC, I-MEN, and S-DSP (STAGE$_1$) shows the best performance among the 16 STAGE candidate combinations, and we choose the best one as our final approach **STAGE**. Compared to BERT without DA, BERT with **STAGE** shows an average accuracy increase of 6.6%. Especially, when there is only 1%, 2% of the training dataset available, BERT with **STAGE** shows an improvement of 47.0%, 10.3% over BERT without DA, respectively.

Even STAGE$_{16}$ demonstrates performance improvement over BERT without DA. It improves the performance by 5.1% compared to BERT without DA. It indicates that other combinations are also effective for DA.

6. Experimental Results

6.1. Experimental Setup

We evaluate the performance of **STAGE** in various environments. The experiments are constructed with six datasets for text classification tasks. We choose six datasets with different sizes and domains: SST-2 (Socher et al., 2013), CR (Hu and Liu, 2004; Liu et al., 2015), SUBJ (Pang and Lee, 2004), TREC (Li and Roth, 2002), PC (Ganapathibhotla and Liu, 2008), IMDB (Maas et al., 2011). In Tables 18 and 19 in the Appendix E, we show the description and summary of the datasets.

We use three pre-trained models: BERT (Devlin et al., 2018), DistilBERT (Sanh et al., 2019), RoBERTa (Liu et al., 2019) to verify the effectiveness of **STAGE** for various pre-trained models. We download the pre-trained model weights from HuggingFace's Transformers[1]. To verify the performance of **STAGE** under various data scarcity, we fine-tune the models with 8 subsets. Each of them has 1%, 2%, 5%, 10%, 30%, 50%, 70%, 100% of the original training dataset, respectively. The average values of five runs of each experiment are presented.

6.2. Overall Performance Comparison

Table 7 shows the overall performances of comparative and proposed method. We also compare the four recent baselines. EDA (Wei and Zou, 2019) used simple operations and selected random operands. BT (back-translation) (Edunov et al., 2018a) is a representative method in the text generation. SSMix (Yoon et al., 2021) is a method that considered the importance of tokens when performing DA. AMR-DA (Shou et al., 2022) used simple operations via AMR graphs.

We can see that **STAGE** consistently demonstrates the DA effect regardless of datasets. It shows greater performance improvements as the amount of data becomes smaller. In comparison to recent methods, the average performance is generally superior too. Specifically, while conventional DA methods often show limited effectiveness in full-dataset scenarios, **STAGE** demonstrates its effectiveness by exhibiting remarkable performance improvement even on the full dataset.

We can compare binary classification (SST-2, CR, SUBJ, PC, and IMDB), and multi classification (TREC-c, TREC-f). In particular, it demonstrates

[1] https://huggingface.co/

Method					Training subset size ratio								Average
					1%	2%	5%	10%	30%	50%	70%	100%	
BERT					55.7	77.0	84.5	86.7	89.0	89.7	90.7	91.2	83.1
STAGE$_{16}$	*D-LE*	*R-MC*	*I-LEN*	*S-DSP*	76.2	83.4	86.3	88.4	89.6	90.5	**91.7**	92.0	87.3
STAGE$_{2}$	*D-ME*	*R-RC*	*I-MEN*	*S-SP*	81.2	**84.9**	87.1	89.4	89.6	90.3	90.6	91.6	88.1
STAGE$_{1}$	*D-ME*	*R-MC*	*I-MEN*	*S-DSP*	**81.9**	**84.9**	**87.6**	**89.6**	**90.3**	**90.7**	91.6	**92.1**	**88.6**

Table 6: Performance results of **STAGE** candidates on the SST-2 dataset.

Data	Method	Training subset size ratio								Avg
		1%	2%	5%	10%	30%	50%	70%	100%	
SST-2	BERT	55.7±3.5	77.0±2.1	84.5±0.5	86.7±1.0	89.0±0.2	89.7±0.3	90.7±0.1	91.2±0.0	83.1
	+EDA	77.3±4.1	79.6±4.4	85.2±1.8	85.9±3.1	88.8±1.3	89.9±0.2	90.2±0.5	90.6±0.5	85.9
	+BT	53.3±0.6	64.9±5.5	85.0±1.3	88.1±0.4	89.2±0.5	89.9±0.5	90.9±0.3	91.6±0.2	81.6
	+SSMix	64.4±5.9	77.0±2.6	86.5±1.0	88.3±0.4	89.7±0.3	90.4±0.4	91.0±0.3	91.2±0.1	84.8
	+AMR-DA	81.2±1.3	**87.2±1.0**	87.3±1.0	87.9±0.4	89.5±0.7	89.3±0.3	90.3±0.1	90.5±0.1	87.9
	+STAGE	**81.9±1.6**	84.9±0.4	**87.6±0.5**	**89.6±0.3**	**90.3±0.2**	**90.7±0.2**	**91.6±0.1**	**92.1±0.1**	**88.6**
CR	BERT	63.2±4.9	62.2±5.6	66.5±0.5	68.4±3.5	83.0±1.2	85.0±2.8	84.6±2.1	87.8±0.5	75.1
	+EDA	61.3±3.1	63.8±2.7	73.1±1.5	81.9±1.2	82.9±2.7	83.9±1.5	85.2±0.8	86.7±1.1	77.3
	+BT	63.6±1.9	64.0±1.4	66.9±3.3	77.7±2.0	86.0±1.0	85.6±1.1	87.3±1.2	87.8±1.1	77.3
	+SSMix	64.7±0.5	68.8±0.3	71.7±0.3	76.9±3.0	86.4±0.6	**87.5±1.8**	86.5±0.4	89.1±0.1	79.0
	+AMR-DA	**67.0±2.9**	67.4±1.5	**76.1±1.3**	83.2±0.3	83.6±0.6	83.1±1.0	83.6±1.9	87.8±0.2	79.0
	+STAGE	69.9±5.4	**71.2±2.9**	75.7±2.8	**84.3±1.6**	**87.2±1.0**	**87.5±1.0**	**90.2±0.7**	**90.6±1.2**	**82.1**
SUBJ	BERT	87.2±2.5	91.4±1.1	92.4±1.0	94.4±0.4	95.4±0.3	96.1±0.4	96.1±0.4	96.3±0.5	93.7
	+EDA	92.3±0.4	93.8±0.6	92.6±0.9	94.3±1.6	96.2±0.2	96.0±1.0	96.3±0.2	96.3±0.2	94.7
	+BT	90.4±1.2	93.0±0.0	**94.1±0.4**	**95.6±0.3**	**96.3±0.2**	96.2±0.7	96.2±0.2	96.8±0.0	94.8
	+SSMix	90.6±0.6	92.4±0.0	93.8±0.3	94.8±0.5	95.6±0.0	**96.6±0.1**	**96.7±0.2**	96.8±0.1	94.7
	+AMR-DA	87.5±3.6	81.7±1.7	81.8±1.0	84.8±1.8	87.7±1.3	88.6±0.9	91.2±0.6	92.0±0.1	86.9
	+STAGE	**92.8±0.8**	**94.0±0.3**	**94.1±0.3**	94.8±0.3	95.7±0.1	96.2±0.4	**96.7±0.2**	**97.0±0.1**	**95.2**
TREC-c	BERT	37.2±2.1	46.8±2.2	60.6±5.4	79.4±2.4	93.6±0.3	94.7±0.2	95.3±0.3	95.6±0.3	75.4
	+EDA	45.1±2.5	68.5±3.5	84.6±1.4	87.8±1.1	94.4±0.3	95.4±0.3	95.8±0.5	95.4±0.3	83.3
	+BT	40.4±2.5	57.7±7.4	84.6±0.9	**92.0±0.4**	94.3±0.0	95.4±0.2	96.0±0.1	96.4±0.3	82.1
	+SSMix	28.4±9.5	52.3±11.4	64.7±4.5	88.0±3.8	**94.9±0.3**	**96.5±0.2**	**96.4±0.3**	**97.0±0.1**	77.3
	+AMR-DA	37.4±1.3	63.2±3.1	78.1±1.1	88.4±3.8	90.8±1.2	93.2±1.1	94.6±0.5	96.4±0.2	80.3
	+STAGE	**45.7±4.5**	**70.5±1.7**	**84.9±2.1**	91.1±0.3	94.3±0.5	95.3±0.3	**96.4±0.1**	96.6±0.2	**84.4**
TREC-f	BERT	1.5±1.5	16.4±6.1	13.5±1.7	41.9±4.7	64.2±1.2	74.4±0.3	78.2±0.9	82.3±0.5	46.6
	+EDA	34.4±1.1	42.5±1.4	54.5±1.1	71.5±0.7	80.0±1.2	84.5±0.6	87.7±0.6	87.3±0.4	67.8
	+BT	18.1±4.5	13.5±1.6	35.2±7.7	55.4±0.2	73.8±1.2	82.4±0.5	85.4±0.8	89.0±0.1	56.6
	+SSMix	25.5±5.2	28.7±9.4	41.8±2.7	56.9±1.2	77.9±0.4	83.7±0.8	86.3±0.5	81.6±0.3	60.3
	+AMR-DA	16.2±6.7	36.3±2.2	49.9±1.6	57.9±1.7	74.6±0.3	77.6±1.1	82.0±0.9	86.7±0.2	60.2
	+STAGE	**38.2±5.1**	**56.5±4.1**	**61.8±1.6**	**74.5±0.9**	**86.4±1.0**	**88.3±0.2**	**90.4±0.3**	**91.2±0.2**	**73.4**
PC	BERT	72.3±1.6	88.2±1.3	91.7±0.2	92.9±0.1	94.0±0.2	94.4±0.2	94.7±0.1	95.2±0.1	90.4
	+EDA	88.4±0.8	90.8±1.2	**92.8±0.5**	92.9±0.5	94.1±0.9	94.2±0.0	92.9±0.5	95.1±0.2	92.6
	+BT	89.9±0.2	90.8±0.0	91.5±0.5	92.2±0.0	94.0±0.0	94.3±0.3	94.6±0.1	95.1±0.1	92.8
	+SSMix	90.4±0.2	**91.5±0.4**	92.7±0.1	93.5±0.0	94.5±0.2	**94.9±0.2**	**95.2±0.1**	95.2±0.1	**93.5**
	+AMR-DA	90.0±0.3	90.8±0.1	92.0±0.4	93.3±0.0	94.0±0.0	94.4±0.0	94.5±0.1	95.3±0.2	93.0
	+STAGE	**91.0±0.3**	91.4±0.2	92.2±0.1	**93.6±0.2**	**94.7±0.2**	94.6±0.2	**95.2±0.2**	**95.4±0.2**	**93.5**
IMDB	BERT	86.9±0.6	85.9±1.1	87.2±0.6	90.0±0.2	91.6±0.0	92.0±0.1	91.6±0.1	92.3±0.1	89.7
	+EDA	86.0±0.6	87.1±1.0	**89.9±0.1**	**90.8±0.5**	91.2±0.2	91.4±0.1	92.2±0.1	93.0±0.1	90.2
	+BT	87.1±0.3	89.0±0.3	89.6±0.6	90.2±0.2	91.4±0.1	91.1±0.0	**92.7±0.0**	93.0±0.1	90.5
	+SSMix	74.3±1.7	82.8±0.4	84.2±0.2	85.4±0.1	86.9±0.3	87.6±0.0	88.2±0.1	88.3±0.1	84.7
	+AMR-DA	86.3±0.6	84.4±2.4	89.6±0.0	90.5±0.6	91.1±0.1	91.9±0.3	92.3±0.0	92.0±0.2	89.8
	+STAGE	**88.2±0.2**	**89.2±0.2**	89.6±0.2	90.6±0.0	**91.8±0.0**	**92.1±0.3**	**92.7±0.3**	**93.1±0.1**	**90.9**

Table 7: **STAGE** performance on various datasets.

superior performance in multi-classification scenarios compared to binary classification situations. In the binary classifications, the overall performance is improved by 1.3% to 7.8% compared models without DA. However, in the multi classifications, our method shows substantial improvements by 11.9% to 35.0%. This shows that our proposed method is very effective for the classification tasks.

There is less performance improvement on the IMDB dataset. The IMDB dataset consists of a large number of significantly long samples compared to other datasets. This characteristic provides the model with an sufficient information for training. Therefore, in Table 7, it is evident that all augmentation methods, including our method, are relatively less effective. Specifically, methods other than ours exhibit performance degradation even in situations with a low data ratio. Despite the chal-

Data	Method	Training subset size ratio				Avg
		1%	10%	50%	100%	
SST-2	DistilBERT	50.4	86.4	88.3	89.8	78.7
	+STAGE	**80.4**	**87.6**	**90.0**	**90.9**	**87.2**
	RoBERTa	50.0	91.8	93.3	94.4	82.4
	+STAGE	**85.1**	**92.9**	**94.3**	**95.2**	**91.9**
CR	DistilBERT	66.1	68.2	**85.2**	84.7	76.1
	+STAGE	**73.8**	**82.0**	84.7	**89.4**	**82.5**
	RoBERTa	64.1	62.1	89.2	89.4	76.2
	+STAGE	**72.1**	**85.2**	**90.5**	**91.4**	**84.8**
SUBJ	DistilBERT	75.2	91.9	94.9	95.8	89.5
	+STAGE	**88.8**	**93.9**	**95.6**	**96.0**	**93.6**
	RoBERTa	73.8	93.4	95.7	96.3	89.8
	+STAGE	**91.3**	**94.3**	**95.8**	**96.4**	**94.5**
TREC-c	DistilBERT	28.1	84.7	**94.5**	95.1	75.6
	+STAGE	**43.8**	**88.3**	94.3	**96.1**	**80.6**
	RoBERTa	31.7	88.6	95.1	**96.2**	77.9
	+STAGE	**44.1**	**90.4**	**95.7**	95.7	**81.5**
TREC-f	DistilBERT	26.2	48.4	72.8	82.7	57.5
	+STAGE	**32.5**	**69.8**	**85.5**	**87.8**	**68.9**
	RoBERTa	4.7	54.5	80.5	87.4	56.8
	+STAGE	**27.5**	**66.8**	**86.1**	**89.9**	**67.6**
PC	DistilBERT	65.4	92.6	93.7	94.3	86.5
	+STAGE	**89.7**	**92.8**	**94.0**	**94.8**	**92.8**
	RoBERTa	68.7	93.0	94.2	94.8	87.7
	+STAGE	**89.5**	**93.5**	**94.7**	**95.1**	**93.2**
IMDB	DistilBERT	86.4	89.3	**91.5**	91.8	89.8
	+STAGE	**86.9**	**89.7**	91.4	**92.1**	**90.0**
	RoBERTa	90.4	**92.7**	93.4	94.1	92.7
	+STAGE	**91.4**	92.4	**93.5**	**94.2**	**92.9**

Table 8: **STAGE** performance on various LMs

lenge of obtaining significant augmentation effects in such a dataset, our approach demonstrates superior effectiveness, establishing it as a sufficiently effective method.

6.3. Performance with Other LMs

We also evaluate the performance with DistilBERT (Sanh et al., 2019) and RoBERTa (Liu et al., 2019). DistilBERT is shallower and has fewer parameters compared to BERT. RoBERTa is larger and trained with more data than BERT. Table 8 shows overall performance on each dataset. It shows that **STAGE** methods is also effective to other LMs.

In Tables 20 and 21 in the Appendix F, we present the whole experimental results compared with baseline methods. It also demonstrate the superiority and applicability of our proposed method.

6.4. Computational Cost Analysis

To generate a co-graph, we scan the words within a window size w for each word in a sentence to count the co-occurrences. If we denote the number of words in a sentences as s and the number of sentences in the dataset as n, the time complexity is $O(nsw)$. Since w is a constant or relatively small compared to s, we may say it is $O(ns)$. If we simply scan the whole dataset, we can build the co-graph. The co-occurrences need to be stored in $v \times v$ space, where v is the total number of corpus. The worse case of the space complexity is $O(v^2)$. However, due to the high sparsity of the co-occurrence matrix, it should require minimal space.

We extract the necessary data in a dictionary format, which includes the number of edges and connected pairs for reduce computational cost. For each sentence, the words existing in the co-graph are used as key. The number of edges in the co-graph and the pair words are used as values. Since the dictionary has a maximum size equal to the number of words s, it reduces searching space. For *D-ME* and *R-MC*, it need retrieve the word with the most number of edges. The time cost for retrieving the word is $O(ns)$, deleting a word is $O(1)$, and replacing connected words is $O(nv)$. The respective time complexities is $O(ns)$ and $O(nv)$. For *I-MEN* and *S-DSP*, it need retrieve the pair. The time cost for retrieving the pair is $O(ns)$, inserting connected words is $O(nv)$, and swapping pair is $O(1)$. The respective time complexities is $O(nv)$ and $O(ns)$ too. We confirm that time complexity has a linear relationship with the size of the data. It is more cost-effective than using DL like BT (Edunov et al., 2018a) and SSMix (Yoon et al., 2021).

τ	Ratio	1%	10%	50%	100%	avg
2	Acc	79.5	88.5	90.0	91.3	87.3
	N_{aug}	300	3,499	17,497	34,596	-
5	Acc	79.4	88.8	90.1	91.5	87.5
	N_{aug}	298	3,487	17,465	34,557	-
10	Acc	**81.9**	**89.6**	90.7	**92.1**	**88.5**
	N_{aug}	294	3,476	17,440	34,511	-
20	Acc	76.5	88.7	90.0	91.6	86.7
	N_{aug}	259	3,453	17,407	34,457	-
30	Acc	72.2	88.5	**90.8**	92.0	85.8
	N_{aug}	183	3,446	17,367	34,422	-

Table 9: Performance comparison with various τ.

6.5. Discussion

6.5.1. How τ affects the performance?

We use a hyperparameter, τ, to generate the co-graph. We analyze how τ affects the performance. Table 9 shows the accuracy and the number of augmented samples, N_{aug}. A smaller value of τ leads to a more edges and inclusion of noisy information. It results in performance degradation due to unreliable information from the co-graph. When τ increases, the co-graph has fewer edges, and finding suitable operands becomes more challenging due to sparsity. It leads to a smaller value of N_{aug}, and the performance decreases. In Table 22 in the Appendix G, we present the whole results on 8 subsets.

Ratio	0.1%		0.3%		1%	
Data	**GPT3MiX**	**STAGE**	**GPT3MiX**	**STAGE**	**GPT3MiX**	**STAGE**
SST-2	78.0	69.5	84.9	85.1	87.7	87.6
CR	70.0	68.7	80.8	80.9	84.7	85.7
SUBJ	85.4	84.9	87.5	91.1	90.6	92.7
TREC-c	47.7	47.5	57.8	54.6	60.5	68.6

Table 10: Performance comparison with LLM-based method

Method		Training subset size ratio				Average
		1%	**10%**	**50%**	**100%**	
BERT		55.7	86.7	89.7	91.2	83.1
STAGE	DS	77.8	89.3	90.2	91.6	87.6
	DR	77.8	88.9	90.3	91.6	87.6
	DI	75.0	88.4	90.0	91.6	86.9
	SR	75.5	88.3	90.0	91.5	86.9
	SI	76.8	89.0	90.1	91.6	87.4
	RI	77.0	89.3	90.0	91.6	87.3
	DSR	78.5	89.0	90.5	91.6	87.8
	DSI	78.1	89.2	90.4	91.8	87.6
	DRI	77.4	89.1	90.5	91.6	87.7
	SRI	78.1	89.4	90.5	91.6	87.6
	DSRI	**81.9**	**89.6**	**90.7**	**92.1**	**88.6**

Table 11: Performance on various combinations

6.5.2. Is it as good as LLM-based methods?

We compare our method with an LLM-based data augmentation, GPT3Mix (Yoo et al., 2021). We adopt the experimental configuration employed in GPT3Mix. We use 0.1%, 0.3%, and 1% of the training set. The number of data augmentation and epochs are set to 10 and 20, respectively. Table 10 shows the experimental results. When the dataset ratio is 0.1%, **STAGE** achieves comparable performance to GPT3Mix except SST-2. In the cases of 0.3% and 1.0%, **STAGE** exhibits superior performance the GPT3Mix. However, GPT3Mix is not cost-effective due to the number of parameters. **STAGE** demonstrates a comparable or superior performance when compared to GPT3Mix, and it is cost-effective, which is a strong advantage.

6.5.3. Are all four operations necessary?

We combine the four operations for **STAGE**. To verify that all the four operations are necessary for the performance improvement, we also evaluate partial combinations of the operations. The experimental results are presented in Table 23. From the experiments, we may conclude that all the operations contribute the performance improvements, which indicates that our four modifications are necessary. In Table 23 in the Appendix H, we present the whole results on 8 subsets.

7. Conclusions

We presented a data augmentation method, **STAGE**, that combined various operations and graph data information. **STAGE** has shown the effectiveness regardless of PLMs or datasets. In particular, it was very effective when the data was insufficient and was on the multiple classification.

8. Acknowledgements

This work was supported in part by the Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2019-0-00421, AI Graduate School Support Program, Sungkyunkwan University; and IITP-2023-2020-0-01821, the ICT Creative Consilience Program).

9. Bibliographical References

Jiaao Chen, Derek Tam, Colin Raffel, Mohit Bansal, and Diyi Yang. 2023. An empirical survey of data augmentation for limited data learning in nlp. *Transactions of the Association for Computational Linguistics*, 11:191–211.

Jiaao Chen, Zichao Yang, and Diyi Yang. 2020. Mixtext: Linguistically-informed interpolation of hidden space for semi-supervised text classification. *arXiv preprint arXiv:2004.12239*.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

Andrew M Dai and Quoc V Le. 2015. Semi-supervised sequence learning. *Advances in neural information processing systems*, 28.

Haixing Dai, Zhengliang Liu, Wenxiong Liao, Xiaoke Huang, Yihan Cao, Zihao Wu, Lin Zhao, Shaochen Xu, Wei Liu, Ninghao Liu, et al. 2023. Auggpt: Leveraging chatgpt for text data augmentation. *arXiv preprint arXiv:2302.13007*.

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*.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. *Advances in neural information processing systems*, 32.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018a. Understanding back-translation at scale. *arXiv preprint arXiv:1808.09381*.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018b. Understanding back-translation at scale. *arXiv preprint arXiv:1808.09381*.

Murthy Ganapathibhotla and Bing Liu. 2008. Mining opinions in comparative sentences. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 241–248.

Hongyu Guo, Yongyi Mao, and Richong Zhang. 2019. Augmenting data with mixup for sentence classification: An empirical study. *arXiv preprint arXiv:1905.08941*.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177.

Akbar Karimi, Leonardo Rossi, and Andrea Prati. 2021. Aeda: An easier data augmentation technique for text classification. *arXiv preprint arXiv:2108.13230*.

Myeongsup Kim and Pilsung Kang. 2022. Text embedding augmentation based on retraining with pseudo-labeled adversarial embedding. *IEEE Access*, 10:8363–8376.

Sosuke Kobayashi. 2018. Contextual augmentation: Data augmentation by words with paradigmatic relations. *arXiv preprint arXiv:1805.06201*.

Oleksandr Kolomiyets, Steven Bethard, and Marie-Francine Moens. 2011. Model-portability experiments for textual temporal analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies*, volume 2, pages 271–276. ACL; East Stroudsburg, PA.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2020. Data augmentation using pre-trained transformer models. *arXiv preprint arXiv:2003.02245*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Sangwon Lee, Ling Liu, and Wonik Choi. 2021. Iterative translation-based data augmentation method for text classification tasks. *IEEE Access*, 9:160437–160445.

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

Qian Liu, Zhiqiang Gao, Bing Liu, and Yuanlin Zhang. 2015. Automated rule selection for aspect extraction in opinion mining. In *Twenty-Fourth international joint conference on artificial intelligence*.

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. *arXiv preprint arXiv:1907.11692*.

Jiawei Luo, Mondher Bouazizi, and Tomoaki Ohtsuki. 2021. Data augmentation for sentiment analysis using sentence compression-based seqgan with data screening. *IEEE Access*, 9:99922–99931.

Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 142–150.

Nikolaos Malandrakis, Minmin Shen, Anuj Goyal, Shuyang Gao, Abhishek Sethi, and Angeliki Metallinou. 2019. Controlled text generation for data augmentation in intelligent artificial agents. *arXiv preprint arXiv:1910.03487*.

E Matthew. 2018. Peters, mark neumann, mohit iyyer, matt gardner, christopher clark, kenton lee, luke zettlemoyer. deep contextualized word representations. In *Proc. of NAACL*.

Zhengjie Miao, Yuliang Li, Xiaolan Wang, and Wang-Chiew Tan. 2020. Snippext: Semi-supervised opinion mining with augmented data. In *Proceedings of The Web Conference 2020*, pages 617–628.

Tong Niu and Mohit Bansal. 2018. Adversarial over-sensitivity and over-stability strategies for dialogue models. *arXiv preprint arXiv:1809.02079*.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. *arXiv preprint cs/0409058*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding with unsupervised learning.

Gözde Gül Şahin. 2022. To augment or not to augment? a comparative study on text augmentation techniques for low-resource nlp. *Computational Linguistics*, 48(1):5–42.

Gözde Gül Şahin and Mark Steedman. 2019. Data augmentation via dependency tree morphing for low-resource languages. *arXiv preprint arXiv:1903.09460*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Improving neural machine translation models with monolingual data. *arXiv preprint arXiv:1511.06709*.

Ziyi Shou, Yuxin Jiang, and Fangzhen Lin. 2022. Amr-da: Data augmentation by abstract meaning representation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3082–3098.

Catarina Silva and Bernardete Ribeiro. 2003. The importance of stop word removal on recall values in text categorization. In *Proceedings of the International Joint Conference on Neural Networks, 2003.*, volume 3, pages 1661–1666. IEEE.

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.

Jason Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196*.

Xing Wu, Chaochen Gao, Meng Lin, Liangjun Zang, Zhongyuan Wang, and Songlin Hu. 2022. Text smoothing: Enhance various data augmentation methods on text classification tasks. *arXiv preprint arXiv:2202.13840*.

Xing Wu, Yibing Liu, Xiangyang Zhou, and Dianhai Yu. 2020a. Distilling knowledge from pre-trained language models via text smoothing. *arXiv preprint arXiv:2005.03848*.

Xing Wu, Shangwen Lv, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019. Conditional bert contextual augmentation. In *Computational Science–ICCS 2019: 19th International Conference, Faro, Portugal, June 12–14, 2019, Proceedings, Part IV 19*, pages 84–95. Springer.

Zhuofeng Wu, Sinong Wang, Jiatao Gu, Madian Khabsa, Fei Sun, and Hao Ma. 2020b. Clear: Contrastive learning for sentence representation. *arXiv preprint arXiv:2012.15466*.

Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. 2020. Unsupervised data augmentation for consistency training. *Advances in neural information processing systems*, 33:6256–6268.

Kang Min Yoo, Dongju Park, Jaewook Kang, Sang-Woo Lee, and Woomyeong Park. 2021. Gpt3mix: Leveraging large-scale language models for text augmentation. *CoRR*, abs/2104.08826.

Soyoung Yoon, Gyuwan Kim, and Kyumin Park. 2021. Ssmix: Saliency-based span mixup for text classification. *arXiv preprint arXiv:2106.08062*.

Shujuan Yu, Jie Yang, Danlei Liu, Runqi Li, Yun Zhang, and Shengmei Zhao. 2019. Hierarchical data augmentation and the application in text classification. *IEEE Access*, 7:185476–185485.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2019. Dialogpt: Large-scale generative pre-training for conversational response generation. *arXiv preprint arXiv:1911.00536*.

Zhengli Zhao, Dheeru Dua, and Sameer Singh. 2017. Generating natural adversarial examples. *arXiv preprint arXiv:1710.11342*.

Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. 2020. Random erasing data augmentation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 13001–13008.

A. Importance of Operand Selection

We present more experiments to examine the importance of operand selection. We use ‘Delete’ and ‘Swap’ operations in this section. In delete operation, we use TF (Term Frequency) score to select different operands. In swap operation, we use similarity score. We evaluate classification ability by varying the operand based on the score. We use the SST-2 dataset and accuracy score as our performance metric.

Figure 3 shows the augmentation results of the delete operation. To delete words in the sentence, we choose the operands based on TF score in the corpus. First, we evaluate TF score of all the words, and choose the words to delete in the order of scores in the sentence. Figure shows the relative performance with respect to the TF score of the deletion word. In the result, the performance varies much depending on which word is chosen for deletion utilizing the same operation. We observe that deleting words with 4th TF scores is the best performance, while deleting words below a certain rank results in decreased performance. It shows the operand selection is crucial.

Figure 4 shows the augmentation results of the swap operation. In the swap operation, we choose a pair of words to swap based on similarity. We first evaluate similarities of all word pairs in the sentence, and choose the pair to swap in order of similarity scores. Figure shows the relative performance with respect to the similarity of the pair. The performance varies much depending on which pair is chosen too. When a word is replaced by a word with a similarity rank of 10, the augmentation performances are the best. We can see that the similarity information and the relation information between words have a significant impact. It is a similar patterns with other modification operations.

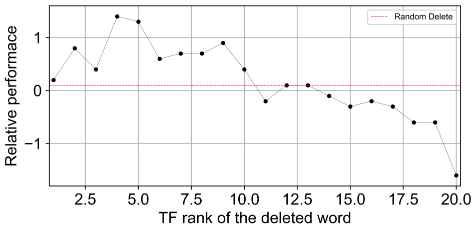

Figure 3: Augmentation result with the delete operation. We repeat augmentation experiments by choose a word to delete in the order of TF (term frequency) in the sentence. It shows that operand selection significantly impacts performance.

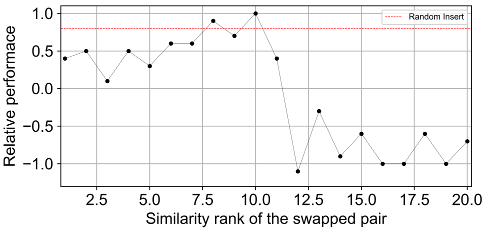

Figure 4: Augmentation result with the swap operation. We repeat augmentation experiments by choose a pair of words to swap in the order of similarity in the sentence. The perforamcne is the best when the 10th similar pair are swapped. It shows that operand selection impacts performance.

B. Description of Proposed Methods

We suggest a various way to choosen operands based on the co-graph for each operation. Table 12 shows the description of proposed methods. It explains each operation abbreviation and how it modifies the sentence.

Operation	Method	Explanation
Delete	*D-RE* (Delete-Random Edge-word)	Delete the random edge-word in the sentence
	D-RI (Delete-Random Isolated-word)	Delete the random isolated-word in the sentence
	D-ME (Delete-Most Edge-word)	Delete the edge-word with the most edges in the sentence
	D-LE (Delete-Least Edge-word)	Delete the edge-word with the least edges in the sentence
Replace	*R-RC* (Replace-Random-Connected)	Replace random word with connected word in the graph
	R-RS (Replace-Random -Similar)	Replace random word with the most similar word in the graph
	R-RDS (Replace-Random-Dissimilar)	Replace random word with the most dissimilar word in the graph
	R-MC (Replace-Most edge-Connected)	Replace the most connected word with connected word in the graph
	R-MS (Replace-Most edge-Similar)	Replace the most connected word with the most similar word in the graph
	R-MDS (Replace-Most edge-Dissimilar)	Replace the most connected word with the most dissimilar word in the graph
Insert	*I-EE* (Insert-Edge-word Expansion)	Insert neighboring word connected to a randomly selected edge-word in the sentence
	I-REN (Insert-Random Edge-word's Neighbors)	Insert random neighboring word connected to edge-words in the sentence
	I-MEN (Insert-Most Edge-word's Neighbors)	Insert most neighboring word connected to edge-words in the sentence
	I-LEN (Insert-Least Edge-word's Neighbors)	Insert least neighboring word connected to edge-words in the sentence
Swap	*S-RP* (Swap-Random Pair)	Swap random word pair connected in the sentence
	S-SP (Swap-Similar Pair)	Swap the most similar word pair in the sentence
	S-DSP (Swap-Dissimilar Pair)	Swap the most dissimilar word pair in the sentence

Table 12: Proposed augmentation methods

C. Evaluation of Proposed Methods for Each Operation

Due to the page limit, we partially present the evaluation result of the proposed methods for each operation. In this section, we present the details of evaluations with the SST-2 dataset. The experiments are conducted in 8 subsets of each training dataset.

C.1. Detail evaluation result of delete

Table 13 shows the performance of delete operation methods. As a result of the experiment, it can be showed that regardless of the size of the data-set, there is a performance improvement after DA compared to before. Furthermore, it has been proven to be more effective in situations where there is limited data, similar to a previous DA approach. We define methods based on different criteria for selecting operands. Through experimental results, we can observe the differences in performance among these methods. When data is insufficient, it is effective to simply remove frequently used words such as stop-words. When data is sufficient, it is better to learn noise sentence by removing frequently used words in some special patterns.

In previous methods, the performance is better when using TF-IDF score or Word2Vec representation than using D-Random. From the perspective of frequency, D-TF-IDF high deletes word that is important in data-set. As a result, it can be seen that deleting important words is meaningful. This method, similar to the one we proposed, utilize term frequency and show similar effects in the results. However, our proposed approach using co-graph considers not only the simple term frequency but also the relational information leading to better performance. From a semantic perspective, the D-Word2Vec deletes words that are similar to the original sentence. However, this approach shows significantly lower effectiveness compared to the term frequency-based method.

C.2. Detail evaluation result of replace

We try to select operands with high grammatical, semantic, and idiomatic correlation with existing word using co-graph. The meaning of being connected by a graph is highly likely to be a word with a high correlation. Table 14 shows the performance of replace operation methods. In the case of a small

Method	Training subset size ratio								Average
	1%	2%	5%	10%	30%	50%	70%	100%	
BERT	55.7	77.0	84.5	86.7	89.0	89.7	90.7	91.2	83.1
D-Random	70.3	80.7	85.1	86.8	87.5	89.1	90.1	90.8	85.1
D-TF-IDF high	75.3	82.3	85.2	87.9	88.4	89.5	89.5	90.4	86.1
D-Word2Vec	72.7	78.1	84.6	87.5	88.5	89.3	90.6	90.9	85.3
D-RE(ours)	74.4	82.3	85.5	88.3	89.1	89.9	90.6	91.3	86.4
D-RI(ours)	70.4	78.4	84.9	87.2	87.7	89.2	89.1	90.9	84.7
D-ME(ours)	**75.4**	**83.1**	**86.2**	**88.4**	89.5	90.1	**91.0**	91.4	**86.9**
D-LE(ours)	74.1	81.8	85.6	88.2	**89.9**	**90.2**	90.5	**91.6**	86.5

Table 13: Performance results of delete methods on SST-2 dataset

Method	Training subset size ratio								Average
	1%	2%	5%	10%	30%	50%	70%	100%	
BERT	55.7	77.0	84.5	86.7	89.0	89.7	90.7	91.2	83.1
R-Random	70.6	81.0	84.8	86.1	87.7	89.1	89.5	89.9	84.8
R-Word2Vec	70.6	79.9	85.4	87.1	88.6	89.5	90.4	90.5	85.3
R-RC(ours)	**79.6**	**84.0**	**86.4**	87.8	89.0	89.4	90.1	90.2	**87.1**
R-RS(ours)	77.5	82.4	85.9	87.6	88.6	89.6	89.8	90.8	86.5
R-RDS(ours)	74.3	80.0	85.0	86.8	89.2	89.6	90.5	91.1	85.8
R-MC(ours)	76.2	83.3	86.0	88.0	**89.6**	90.4	**91.1**	**91.7**	87.0
R-MS(ours)	76.0	83.1	85.9	**88.1**	89.1	89.9	90.3	91.6	86.8
R-MDS(ours)	72.5	83.0	85.7	88.0	89.5	**90.5**	90.4	91.6	86.4

Table 14: Performance results of replace methods on SST-2 dataset

Method	Training subset size ratio								Average
	1%	2%	5%	10%	30%	50%	70%	100%	
BERT	55.7	77.0	84.5	86.7	89.0	89.7	90.7	91.2	83.1
I-Random	73.5	81.4	84.8	86.9	88.9	89.6	90.3	90.8	85.8
I-Word2Vec	71.7	81.4	85.7	87.3	89.2	90.0	90.2	91.0	85.8
I-EE(ours)	75.7	82.2	86.1	87.7	89.5	90.1	90.6	91.4	86.7
I-REN(ours)	77.2	82.2	86.6	88.4	89.6	90.2	91.2	91.6	87.1
I-MEN(ours)	**79.5**	**82.8**	**86.9**	**89.2**	89.2	90.2	91.3	**91.7**	**87.6**
I-LEN(ours)	78.1	82.3	86.8	88.5	**89.9**	**90.6**	**91.5**	91.6	87.4

Table 15: Performance results of insert methods on SST-2 dataset

corpus, *R-RC* shows better results than similarity-based ones, and the the case of a large corpus, *R-MC* shows better than similarity-based ones. It can be noted that choose substitute words by the co-graph is better than by similarity. *R-RC* and *R-MC* are differ in the criteria for choosing what words in the sentence. *R-MC* chooses a operand that has the highest correlation with other words in the sentence. This significantly alters the meaning of the sentence and shows better performance when data-set size ratio is large.

R-Word2Vec is expected to select synonym and preserve the meaning of the original sentence. However, R-Word2Vec sometimes shows lower performance than R-Random when the data set-size ratio is small. This is considered to be because there are not enough synonym words in the vocabulary. Searching for synonym words is meaningless when it doesn't have a vocabulary of sufficient size. Similarly *R-RS*, *R-RDS*, *R-MS* and *R-MDS* are not very useful because vocabulary size is too small to find the synonym or antonym.

C.3. Detail evaluation result of insert

Table 15 shows the performance of insert operation methods. *I-EE* utilizes the co-graph to select operands that have a relation with the edge-word in the sentence. This method is more effective compared to the simple similarity-based approaches like I-Word2Vec or the base method I-Random.

In *I-EE*, all words connected to the specific edge-word become candidates for operands. *I-EE* is highly influenced by the specific edge word that is selected. On the other hand, methods like *I-REN*, *I-MEN*, and *I-LEN* consider all edge-words in sentence and narrow down the candidate range for operands. In table *I-REN*, it is less influenced by the choice of edge-word and consistently shows better performance as a result. Furthermore, *I-MEN* may select highly related word with sentence. *I-LEN* may select unrelated one. Through this, we confirm that finding words related to the entire sentence is more crucial in the insert operations, rather than focusing on a single word and its related words.

Method	Training subset size ratio								Average
	1%	2%	5%	10%	30%	50%	70%	100%	
BERT	55.7	77.0	84.5	86.7	89.0	89.7	90.7	91.2	83.1
S-Random	72.3	79.5	85.4	87.5	88.0	89.3	89.4	90.2	85.2
S-Word2Vec	72.5	80.3	85.3	87.2	88.2	90.0	90.1	90.5	85.5
S-RP	77.7	83.7	85.9	87.7	**89.2**	90.0	90.4	90.4	86.9
S-SP	**79.8**	**83.8**	**86.1**	87.6	88.1	89.8	89.6	90.3	**86.9**
S-DSP	72.3	81.8	85.8	**87.8**	88.9	**90.1**	**90.8**	**91.5**	86.1

Table 16: Performance results of swap methods on SST-2 dataset

Method					Training subset size ratio								Average
					1%	2%	5%	10%	30%	50%	70%	100%	
BERT					55.7	77.0	84.5	86.7	89.0	89.7	90.7	91.2	83.1
STAGE$_{16}$	*D-LE*	*R-MC*	*I-LEN*	*S-DSP*	76.2	83.4	86.3	88.4	89.6	90.5	**91.7**	92.0	87.3
STAGE$_{15}$	*D-LE*	*R-RC*	*I-MEN*	*S-SP*	78.8	83.5	86.5	88.4	89.5	90.3	90.9	91.4	87.4
STAGE$_{14}$	*D-LE*	*R-MC*	*I-LEN*	*S-SP*	78.5	83.7	87.3	88.5	89.9	90.3	90.7	91.3	87.5
STAGE$_{13}$	*D-ME*	*R-RC*	*I-MEN*	*S-SP*	78.5	84.0	87.2	88.4	89.8	90.2	90.8	92.0	87.6
STAGE$_{12}$	*D-LE*	*R-RC*	*I-MEN*	*S-DSP*	78.4	83.9	86.7	89.4	89.6	90.4	91.3	91.8	87.7
STAGE$_{11}$	*D-LE*	*R-RC*	*I-LEN*	*S-SP*	78.7	**84.6**	87.1	89.0	89.8	90.1	90.8	91.8	87.7
STAGE$_{10}$	*D-ME*	*R-RC*	*I-MEN*	*S-DSP*	79.4	84.4	86.8	88.6	90.1	90.2	91.0	92.0	87.8
STAGE$_{9}$	*D-LE*	*R-MC*	*I-LEN*	*S-SP*	78.6	84.8	87.3	88.8	**90.4**	90.0	91.0	91.4	87.8
STAGE$_{8}$	*D-ME*	*R-RC*	*I-MEN*	*S-DSP*	80.0	83.9	86.9	89.0	89.8	90.5	90.6	91.6	87.8
STAGE$_{7}$	*D-ME*	*R-MC*	*I-LEN*	*S-SP*	79.4	83.7	87.5	89.1	89.5	90.2	90.9	**92.1**	87.8
STAGE$_{6}$	*D-LE*	*R-RC*	*I-MEN*	*S-DSP*	80.0	84.4	86.7	89.0	90.2	90.2	90.9	91.6	87.9
STAGE$_{5}$	*D-LE*	*R-MC*	*I-LEN*	*S-DSP*	79.5	84.2	86.6	89.0	90.2	90.4	91.4	91.7	87.9
STAGE$_{4}$	*D-ME*	*R-MC*	*I-LEN*	*S-DSP*	79.0	84.1	87.0	89.3	90.1	90.5	91.4	91.8	87.9
STAGE$_{3}$	*D-ME*	*R-MC*	*I-LEN*	*S-SP*	81.0	84.5	87.2	88.8	89.7	90.2	91.5	91.9	88.1
STAGE$_{2}$	*D-ME*	*R-RC*	*I-MEN*	*S-SP*	81.2	**84.9**	87.1	89.4	89.6	90.3	90.6	91.6	88.1
STAGE$_{1}$	*D-ME*	*R-MC*	*I-MEN*	*S-DSP*	**81.9**	**84.9**	**87.6**	**89.6**	90.3	**90.7**	91.6	**92.1**	**88.6**

Table 17: Performance results of **STAGE** candidates on the SST-2 dataset

C.4. Detail evaluation result of swap

Table 16 shows the performance of swap operation methods. Our proposed methods considering relation information show better performance. In addition, *S-SP*, *S-DSP* consider relation information but also consider semantic similarity. *S-SP* swaps the most similar word pair connected in the graph. It is believed that to preserve the meaning of sentence, and it is effective when data is insufficient. On the other hand, *S-DSP* is believed that to variant the meaning of sentence, and it is effective when data is sufficient.

An interesting result from the swap operation is that the performance of the S-Random is quite good as S-Word2Vec. This means that swapping only with similarity is not effective. We use the co-graph to focus on relation information and it is effective.

D. Superiority of STAGE Candidates

As mentioned in the paper, we select 2 methods for each operation, and combine them to create 16 candidate combinations. In this section, we present the whole comparative results. Table 17 shows the combinations and experimental results for each candidate. Overall, the average accuracy of most combinations are around 88%. All of them show significant performance improvement compared to BERT without DA. The combined ones also show superior performance compared to single-operation results. Through this, we can see that operation combinations are more effective.

Dataset	C	$\mathbf{N_{train}}$	$\mathbf{N_{test}}$	V
SST-2	2	6,920	1,821	11,925
CR	2	1,863	208	3,775
SUBJ	2	8,000	1,000	19,179
TREC-coarse	6	5,452	500	5,353
TREC-fine	50	5,452	500	5,353
PC	2	39,428	4,504	8,666
IMDB	2	20,000	25,000	72,831

Table 18: Summary of six classification datasets. C: number of classes, N_{train}: number of training samples, N_{test}: number of test samples, V: size of vocabulary

E. Description of Datasets

We conduct an experiment using six benchmark data-sets for the text classification task. Table 18 is the detailed information of the data-sets. The detailed descriptions of benchmark data-sets are in the Table 19.

Dataset	Description
SST-2	It is the Stanford Sentiment Treebank(Socher et al., 2013) dataset, which is movie review data. It is binary classification dataset for sentiment analysis.
CR	It is the Customer Reviews(Hu and Liu, 2004; Liu et al., 2015) dataset. It is a binary classification dataset, and it is compared with SST-2 from different domains
SUBJ	The subjectivity(Pang and Lee, 2004) dataset is divided into subjectivity and objectivity for movie reviews. It is similar to sentiment analysis, but what we want to distinguish is the problem of binary classification of data based on subjectivity, not sentiment.
TREC	It is a dataset for question classification problem provided by Text Retrieval Conference(Li and Roth, 2002). The dataset is divided into TREC-coarse and TREC-fine in detail. TREC-coarse consists of 6 comprehensive labels and TREC-fine consists of 50 subdivided labels.
PC	Pro-Con dataset(Ganapathibhotla and Liu, 2008) is a dataset for binary classification of sentences into two classes, pros and cons. Pros means agree and cons means disagree.
IMDB	IMDB(Maas et al., 2011) is the most well-known dataset for movie review datasets. Compared with other movie review datasets, the number of sentences included in the dataset is large and the words used are diverse.

Table 19: Descriptions of benchmark datasets

F. Evaluation of STAGE with other Language Models

In Section 6, we briefly present the comparative results with DistilBERT and RoBERTa due to the page limit. We have performed various comparative evaluations with DistilBERT and RoBERTa.

We apply **STAGE** to the benchmark datasets SST-2, CR, SUBJ, TREC, PC, and IMDB with BERT, DistilBERT, and RoBERTa. To simulate the scarcity of training data, we train models with 8 subsets of the original training dataset like evaluation. We also compare with the four recent baselines as we do in Section 6: EDA (Wei and Zou, 2019), BT (back-translation) (Edunov et al., 2018a), SSMix (Yoon et al., 2021), and AMR-DA (Shou et al., 2022). The experimental results are shown in Tables 20 and 21.

DistillBERT and RoBERTa do not perform well when fine-tune with very small datasets (such as when the data ratio is 1%). Our proposed data augmentation technique, **STAGE**, can easily overcome this issue. Although there is not a strong linear correlation between the amount of data and the performance improvement, there is a positive correlation to some extent. Our approach also show superior performance compared with the four baselines in all the cases. We cocnlude that **STAGE** can be used regardless of the type of pre-trained models.

Data	Method	Training subset size ratio								Avg
		1%	2%	5%	10%	30%	50%	70%	100%	
SST-2	DistilBERT	$50.4_{\pm 5.1}$	$68.2_{\pm 3.7}$	$83.9_{\pm 0.2}$	$86.4_{\pm 0.7}$	$88.2_{\pm 0.3}$	$88.3_{\pm 0.1}$	$89.4_{\pm 0.2}$	$89.8_{\pm 0.1}$	80.6
	+EDA	$76.2_{\pm 3.0}$	$81.4_{\pm 0.5}$	$84.5_{\pm 0.5}$	$86.7_{\pm 0.2}$	$86.4_{\pm 2.1}$	$88.9_{\pm 0.6}$	$90.2_{\pm 0.4}$	$89.7_{\pm 0.2}$	85.5
	+BT	$50.2_{\pm 0.2}$	$54.4_{\pm 2.9}$	$80.1_{\pm 2.4}$	$86.0_{\pm 0.5}$	$88.4_{\pm 0.5}$	$89.1_{\pm 0.4}$	$89.9_{\pm 0.2}$	$\mathbf{90.9}_{\pm \mathbf{0.7}}$	78.6
	+AMR-DA	$79.0_{\pm 1.0}$	$\mathbf{82.4}_{\pm \mathbf{0.3}}$	$84.5_{\pm 0.6}$	$85.8_{\pm 0.2}$	$86.7_{\pm 0.3}$	$87.8_{\pm 0.2}$	$88.6_{\pm 0.0}$	$90.0_{\pm 0.0}$	85.6
	+STAGE	$\mathbf{80.4}_{\pm \mathbf{0.5}}$	$82.2_{\pm 0.3}$	$\mathbf{85.5}_{\pm \mathbf{0.4}}$	$\mathbf{87.6}_{\pm \mathbf{0.2}}$	$\mathbf{89.4}_{\pm \mathbf{0.4}}$	$\mathbf{90.0}_{\pm \mathbf{0.2}}$	$\mathbf{90.3}_{\pm \mathbf{0.1}}$	$\mathbf{90.9}_{\pm \mathbf{0.1}}$	**87.0**
CR	DistilBERT	$66.1_{\pm 2.8}$	$67.8_{\pm 1.8}$	$68.1_{\pm 2.3}$	$68.2_{\pm 2.1}$	$84.1_{\pm 2.6}$	$85.2_{\pm 1.3}$	$84.3_{\pm 1.9}$	$84.7_{\pm 1.4}$	76.1
	+EDA	$65.6_{\pm 0.6}$	$63.9_{\pm 3.8}$	$72.0_{\pm 3.5}$	$79.4_{\pm 2.4}$	$81.7_{\pm 1.5}$	$82.5_{\pm 2.4}$	$82.1_{\pm 0.7}$	$84.2_{\pm 0.4}$	76.4
	+BT	$64.0_{\pm 2.2}$	$66.7_{\pm 1.9}$	$66.4_{\pm 2.4}$	$79.1_{\pm 4.1}$	$82.8_{\pm 0.3}$	$82.4_{\pm 1.1}$	$84.2_{\pm 1.4}$	$85.6_{\pm 0.9}$	76.4
	+AMR-DA	$69.5_{\pm 0.3}$	$70.0_{\pm 4.7}$	$75.4_{\pm 1.1}$	$\mathbf{82.0}_{\pm \mathbf{1.3}}$	$83.3_{\pm 1.6}$	$80.9_{\pm 1.1}$	$80.5_{\pm 0.4}$	$83.7_{\pm 0.2}$	78.2
	+STAGE	$\mathbf{73.8}_{\pm \mathbf{4.4}}$	$\mathbf{70.7}_{\pm \mathbf{4.2}}$	$\mathbf{76.9}_{\pm \mathbf{2.4}}$	$\mathbf{82.0}_{\pm \mathbf{2.2}}$	$\mathbf{83.9}_{\pm \mathbf{1.8}}$	$\mathbf{84.7}_{\pm \mathbf{1.3}}$	$\mathbf{85.4}_{\pm \mathbf{1.4}}$	$\mathbf{89.4}_{\pm \mathbf{1.3}}$	**80.9**
SUBJ	DistilBERT	$75.2_{\pm 6.7}$	$89.0_{\pm 0.2}$	$91.6_{\pm 0.5}$	$91.9_{\pm 0.4}$	$94.3_{\pm 0.3}$	$94.9_{\pm 0.4}$	$95.2_{\pm 0.4}$	$95.8_{\pm 0.3}$	91.0
	+EDA	$\mathbf{89.3}_{\pm \mathbf{0.6}}$	$\mathbf{90.4}_{\pm \mathbf{0.8}}$	$91.7_{\pm 0.6}$	$93.9_{\pm 0.3}$	$94.1_{\pm 0.2}$	$94.6_{\pm 0.3}$	$95.5_{\pm 0.2}$	$95.8_{\pm 0.4}$	93.1
	+BT	$88.5_{\pm 0.4}$	$89.6_{\pm 1.3}$	$91.4_{\pm 0.2}$	$\mathbf{94.2}_{\pm \mathbf{0.3}}$	$\mathbf{94.9}_{\pm \mathbf{0.4}}$	$95.5_{\pm 0.5}$	$95.6_{\pm 0.5}$	$95.9_{\pm 0.3}$	93.2
	+AMR-DA	$87.7_{\pm 2.4}$	$82.8_{\pm 1.6}$	$83.9_{\pm 1.2}$	$81.5_{\pm 1.8}$	$85.0_{\pm 1.1}$	$85.2_{\pm 1.2}$	$87.7_{\pm 1.4}$	$89.2_{\pm 0.5}$	85.4
	+STAGE	$88.8_{\pm 1.6}$	$90.3_{\pm 0.5}$	$\mathbf{91.8}_{\pm \mathbf{0.3}}$	$93.9_{\pm 0.1}$	$94.3_{\pm 1.1}$	$\mathbf{95.6}_{\pm \mathbf{0.2}}$	$\mathbf{95.7}_{\pm \mathbf{0.2}}$	$\mathbf{96.0}_{\pm \mathbf{0.3}}$	**93.3**
TREC-c	DistilBERT	$28.1_{\pm 8.4}$	$45.5_{\pm 8.5}$	$56.9_{\pm 2.4}$	$84.7_{\pm 1.8}$	$92.7_{\pm 0.0}$	$94.5_{\pm 0.1}$	$94.8_{\pm 0.4}$	$95.1_{\pm 0.4}$	74.0
	+EDA	$43.5_{\pm 3.7}$	$72.5_{\pm 0.9}$	$82.2_{\pm 0.6}$	$\mathbf{88.8}_{\pm \mathbf{0.7}}$	$92.9_{\pm 0.4}$	$94.2_{\pm 1.0}$	$95.3_{\pm 0.8}$	$95.2_{\pm 0.4}$	83.1
	+BT	$37.0_{\pm 2.1}$	$55.4_{\pm 7.9}$	$82.5_{\pm 1.3}$	$\mathbf{88.8}_{\pm \mathbf{0.7}}$	$\mathbf{93.1}_{\pm \mathbf{0.3}}$	$\mathbf{95.1}_{\pm \mathbf{0.6}}$	$95.3_{\pm 0.4}$	$95.8_{\pm 0.4}$	80.3
	+AMR-DA	$42.8_{\pm 1.1}$	$72.6_{\pm 2.3}$	$82.8_{\pm 0.5}$	$87.5_{\pm 0.1}$	$92.2_{\pm 1.4}$	$93.6_{\pm 0.7}$	$92.2_{\pm 0.8}$	$93.1_{\pm 0.2}$	82.1
	+STAGE	$\mathbf{43.8}_{\pm \mathbf{4.5}}$	$\mathbf{73.5}_{\pm \mathbf{3.2}}$	$\mathbf{83.5}_{\pm \mathbf{2.2}}$	$88.3_{\pm 1.7}$	$93.0_{\pm 0.5}$	$94.3_{\pm 1.0}$	$\mathbf{95.4}_{\pm \mathbf{0.3}}$	$\mathbf{96.1}_{\pm \mathbf{0.6}}$	**83.5**
TREC-f	DistilBERT	$26.2_{\pm 7.0}$	$29.7_{\pm 3.9}$	$33.8_{\pm 1.4}$	$48.4_{\pm 2.7}$	$65.7_{\pm 0.3}$	$72.8_{\pm 1.6}$	$76.9_{\pm 0.8}$	$82.7_{\pm 0.6}$	54.5
	+EDA	$32.2_{\pm 3.6}$	$45.5_{\pm 1.0}$	$\mathbf{62.4}_{\pm \mathbf{0.5}}$	$68.3_{\pm 2.4}$	$80.1_{\pm 2.3}$	$84.1_{\pm 1.2}$	$87.4_{\pm 0.4}$	$87.7_{\pm 0.9}$	68.4
	+BT	$12.9_{\pm 2.2}$	$11.3_{\pm 0.4}$	$49.2_{\pm 4.3}$	$60.9_{\pm 0.2}$	$73.9_{\pm 0.8}$	$81.6_{\pm 1.1}$	$84.1_{\pm 1.1}$	$88.1_{\pm 0.5}$	57.0
	+AMR-DA	$24.6_{\pm 3.0}$	$35.3_{\pm 2.4}$	$48.6_{\pm 2.0}$	$58.5_{\pm 0.6}$	$71.3_{\pm 0.2}$	$74.7_{\pm 2.9}$	$80.6_{\pm 1.8}$	$83.5_{\pm 0.2}$	59.6
	+STAGE	$\mathbf{32.5}_{\pm \mathbf{2.0}}$	$\mathbf{49.1}_{\pm \mathbf{0.2}}$	$60.0_{\pm 2.0}$	$\mathbf{69.8}_{\pm \mathbf{2.7}}$	$\mathbf{80.4}_{\pm \mathbf{1.0}}$	$\mathbf{85.5}_{\pm \mathbf{0.4}}$	$\mathbf{87.7}_{\pm \mathbf{0.6}}$	$\mathbf{87.8}_{\pm \mathbf{0.5}}$	**69.1**
PC	DistilBERT	$65.4_{\pm 4.1}$	$75.0_{\pm 3.2}$	$91.4_{\pm 0.4}$	$92.6_{\pm 0.2}$	$93.5_{\pm 0.1}$	$93.7_{\pm 0.1}$	$94.0_{\pm 0.2}$	$94.3_{\pm 0.1}$	87.5
	+EDA	$89.6_{\pm 0.6}$	$90.9_{\pm 0.3}$	$91.6_{\pm 0.4}$	$\mathbf{92.8}_{\pm \mathbf{0.2}}$	$93.4_{\pm 0.3}$	$94.2_{\pm 0.1}$	$\mathbf{94.3}_{\pm \mathbf{0.2}}$	$94.5_{\pm 0.1}$	92.6
	+BT	$89.5_{\pm 0.4}$	$91.0_{\pm 0.2}$	$\mathbf{92.0}_{\pm \mathbf{0.3}}$	$92.4_{\pm 0.1}$	$93.5_{\pm 0.2}$	$\mathbf{94.3}_{\pm \mathbf{0.1}}$	$94.1_{\pm 0.1}$	$94.7_{\pm 0.1}$	92.6
	+AMR-DA	$89.6_{\pm 0.2}$	$90.9_{\pm 0.7}$	$91.6_{\pm 0.2}$	$92.7_{\pm 0.2}$	$\mathbf{93.6}_{\pm \mathbf{0.1}}$	$94.0_{\pm 0.1}$	$93.9_{\pm 0.2}$	$94.0_{\pm 0.1}$	92.5
	+STAGE	$\mathbf{89.7}_{\pm \mathbf{0.5}}$	$\mathbf{91.1}_{\pm \mathbf{0.3}}$	$91.9_{\pm 0.2}$	$\mathbf{92.8}_{\pm \mathbf{0.2}}$	$\mathbf{93.6}_{\pm \mathbf{0.3}}$	$94.0_{\pm 0.2}$	$\mathbf{94.3}_{\pm \mathbf{0.0}}$	$\mathbf{94.8}_{\pm \mathbf{0.2}}$	**92.8**
IMDB	DistilBERT	$86.4_{\pm 0.4}$	$86.5_{\pm 0.2}$	$87.4_{\pm 0.5}$	$89.3_{\pm 0.5}$	$90.5_{\pm 0.5}$	$91.5_{\pm 0.0}$	$91.7_{\pm 0.2}$	$91.8_{\pm 0.1}$	89.4
	+EDA	$80.7_{\pm 3.2}$	$84.9_{\pm 0.8}$	$87.5_{\pm 0.4}$	$89.2_{\pm 0.0}$	$90.4_{\pm 0.5}$	$91.3_{\pm 0.0}$	$91.8_{\pm 0.1}$	$\mathbf{92.1}_{\pm \mathbf{0.1}}$	88.4
	+BT	$84.5_{\pm 0.7}$	$86.3_{\pm 0.2}$	$87.6_{\pm 0.5}$	$89.1_{\pm 0.1}$	$\mathbf{91.0}_{\pm \mathbf{0.3}}$	$\mathbf{91.7}_{\pm \mathbf{0.1}}$	$\mathbf{92.0}_{\pm \mathbf{0.0}}$	$92.0_{\pm 0.2}$	89.2
	+AMR-DA	$86.4_{\pm 0.5}$	$86.6_{\pm 0.5}$	$86.8_{\pm 1.4}$	$88.3_{\pm 0.9}$	$90.5_{\pm 0.2}$	$90.7_{\pm 0.7}$	$91.8_{\pm 0.2}$	$91.9_{\pm 0.1}$	89.1
	+STAGE	$\mathbf{86.9}_{\pm \mathbf{0.3}}$	$\mathbf{87.1}_{\pm \mathbf{0.3}}$	$\mathbf{88.1}_{\pm \mathbf{0.3}}$	$\mathbf{89.7}_{\pm \mathbf{0.3}}$	$\mathbf{91.0}_{\pm \mathbf{0.3}}$	$91.4_{\pm 0.1}$	$\mathbf{92.0}_{\pm \mathbf{0.2}}$	$\mathbf{92.1}_{\pm \mathbf{0.2}}$	**89.8**

Table 20: **STAGE** performance on DistilBERT

Data	Method	Training subset size ratio								Avg
		1%	2%	5%	10%	30%	50%	70%	100%	
SST-2	RoBERTa	50.0$_{\pm 7.6}$	50.0$_{\pm 5.8}$	89.0$_{\pm 0.8}$	91.8$_{\pm 0.1}$	92.5$_{\pm 0.2}$	93.3$_{\pm 0.2}$	93.6$_{\pm 0.1}$	94.4$_{\pm 0.1}$	81.8
	+EDA	82.8$_{\pm 3.8}$	87.0$_{\pm 1.4}$	89.9$_{\pm 0.7}$	91.3$_{\pm 0.9}$	92.6$_{\pm 0.1}$	93.1$_{\pm 0.1}$	93.0$_{\pm 0.6}$	93.8$_{\pm 0.5}$	90.4
	+BT	53.3$_{\pm 3.8}$	49.9$_{\pm 1.4}$	87.7$_{\pm 0.7}$	90.8$_{\pm 0.9}$	92.6$_{\pm 0.1}$	93.2$_{\pm 0.1}$	93.6$_{\pm 0.6}$	94.0$_{\pm 0.5}$	81.8
	+AMR-DA	84.5$_{\pm 1.5}$	86.0$_{\pm 1.3}$	89.3$_{\pm 1.1}$	91.3$_{\pm 0.0}$	91.4$_{\pm 0.1}$	91.6$_{\pm 0.3}$	92.7$_{\pm 0.0}$	93.0$_{\pm 0.0}$	90.0
	+STAGE	**85.1**$_{\pm 0.3}$	**88.7**$_{\pm 0.1}$	**91.9**$_{\pm 0.2}$	**92.9**$_{\pm 0.2}$	**93.5**$_{\pm 0.2}$	**94.3**$_{\pm 0.2}$	**94.5**$_{\pm 0.1}$	**95.2**$_{\pm 0.1}$	**92.1**
CR	RoBERTa	64.1$_{\pm 3.1}$	66.0$_{\pm 0.1}$	67.2$_{\pm 0.0}$	62.1$_{\pm 2.9}$	89.0$_{\pm 0.3}$	89.2$_{\pm 1.1}$	89.3$_{\pm 1.0}$	89.4$_{\pm 1.7}$	80.9
	+EDA	65.2$_{\pm 2.7}$	70.5$_{\pm 1.5}$	77.7$_{\pm 2.7}$	84.3$_{\pm 2.2}$	88.2$_{\pm 0.8}$	89.4$_{\pm 2.6}$	90.8$_{\pm 0.8}$	90.7$_{\pm 1.4}$	82.1
	+BT	66.7$_{\pm 2.2}$	63.2$_{\pm 1.9}$	64.0$_{\pm 1.4}$	83.4$_{\pm 2.5}$	89.4$_{\pm 0.3}$	90.4$_{\pm 0.5}$	89.1$_{\pm 0.6}$	89.3$_{\pm 0.4}$	79.4
	+AMR-DA	66.4$_{\pm 1.4}$	71.4$_{\pm 3.1}$	**77.8**$_{\pm 2.2}$	82.5$_{\pm 2.2}$	88.2$_{\pm 1.2}$	89.2$_{\pm 0.8}$	90.2$_{\pm 0.4}$	90.3$_{\pm 0.2}$	82.0
	+STAGE	**72.1**$_{\pm 3.0}$	**72.3**$_{\pm 5.3}$	**77.8**$_{\pm 3.1}$	**85.2**$_{\pm 3.5}$	**90.6**$_{\pm 0.6}$	**90.5**$_{\pm 1.9}$	**91.0**$_{\pm 0.6}$	**91.4**$_{\pm 0.7}$	**83.9**
SUBJ	RoBERTa	73.8$_{\pm 4.9}$	92.0$_{\pm 0.5}$	92.8$_{\pm 0.6}$	93.4$_{\pm 0.7}$	95.0$_{\pm 0.7}$	95.7$_{\pm 0.3}$	95.3$_{\pm 0.9}$	96.3$_{\pm 0.6}$	91.8
	+EDA	91.2$_{\pm 0.4}$	92.3$_{\pm 0.0}$	93.0$_{\pm 1.8}$	94.1$_{\pm 0.3}$	**95.2**$_{\pm 0.5}$	95.4$_{\pm 0.3}$	95.1$_{\pm 0.3}$	95.6$_{\pm 0.2}$	93.9
	+BT	76.6$_{\pm 1.7}$	91.5$_{\pm 1.1}$	**93.8**$_{\pm 0.4}$	**95.2**$_{\pm 0.0}$	95.0$_{\pm 0.3}$	95.5$_{\pm 0.0}$	95.6$_{\pm 0.4}$	95.7$_{\pm 0.3}$	92.2
	+AMR-DA	90.3$_{\pm 0.2}$	85.4$_{\pm 2.4}$	85.9$_{\pm 0.0}$	85.7$_{\pm 1.0}$	87.4$_{\pm 1.1}$	89.1$_{\pm 0.6}$	91.7$_{\pm 0.9}$	93.0$_{\pm 0.2}$	88.6
	+STAGE	**91.3**$_{\pm 0.6}$	**92.6**$_{\pm 0.5}$	93.0$_{\pm 1.1}$	94.3$_{\pm 0.3}$	**95.2**$_{\pm 1.1}$	**95.8**$_{\pm 0.4}$	**96.1**$_{\pm 0.3}$	**96.4**$_{\pm 0.1}$	**94.7**
TREC-c	RoBERTa	31.7$_{\pm 2.9}$	27.5$_{\pm 5.8}$	64.5$_{\pm 2.0}$	88.6$_{\pm 2.7}$	93.6$_{\pm 1.0}$	95.1$_{\pm 0.4}$	96.2$_{\pm 0.6}$	96.2$_{\pm 0.3}$	74.2
	+EDA	43.3$_{\pm 3.2}$	76.1$_{\pm 1.2}$	83.6$_{\pm 1.2}$	88.8$_{\pm 0.8}$	**94.3**$_{\pm 0.6}$	94.5$_{\pm 1.0}$	94.8$_{\pm 1.3}$	95.7$_{\pm 0.3}$	83.8
	+BT	18.7$_{\pm 1.0}$	53.9$_{\pm 8.4}$	**91.4**$_{\pm 1.4}$	**92.5**$_{\pm 0.8}$	**94.3**$_{\pm 0.8}$	94.3$_{\pm 0.7}$	95.1$_{\pm 0.3}$	**96.0**$_{\pm 0.1}$	79.5
	+AMR-DA	43.4$_{\pm 2.6}$	73.3$_{\pm 0.7}$	83.9$_{\pm 2.1}$	86.7$_{\pm 0.9}$	91.8$_{\pm 1.7}$	93.5$_{\pm 0.8}$	93.0$_{\pm 1.0}$	94.2$_{\pm 0.2}$	82.5
	+STAGE	**44.1**$_{\pm 4.5}$	**76.6**$_{\pm 1.1}$	84.3$_{\pm 2.4}$	90.4$_{\pm 1.7}$	93.9$_{\pm 0.8}$	**95.7**$_{\pm 0.4}$	**95.9**$_{\pm 0.3}$	95.7$_{\pm 0.1}$	**84.6**
TREC-f	RoBERTa	4.7$_{\pm 3.7}$	20.6$_{\pm 7.1}$	33.8$_{\pm 1.0}$	54.5$_{\pm 1.7}$	73.9$_{\pm 1.1}$	80.5$_{\pm 0.5}$	85.3$_{\pm 0.5}$	87.4$_{\pm 1.1}$	55.1
	+EDA	**32.9**$_{\pm 3.6}$	37.1$_{\pm 1.0}$	55.2$_{\pm 0.5}$	**69.9**$_{\pm 2.4}$	82.6$_{\pm 2.3}$	85.4$_{\pm 1.2}$	86.3$_{\pm 0.4}$	88.6$_{\pm 0.9}$	**67.2**
	+BT	11.0$_{\pm 3.6}$	11.1$_{\pm 4.2}$	55.8$_{\pm 0.5}$	64.0$_{\pm 0.8}$	82.8$_{\pm 0.7}$	85.1$_{\pm 1.3}$	86.9$_{\pm 0.6}$	89.7$_{\pm 0.2}$	60.8
	+AMR-DA	25.9$_{\pm 1.6}$	**44.2**$_{\pm 2.7}$	**58.2**$_{\pm 1.1}$	68.3$_{\pm 0.7}$	77.2$_{\pm 1.4}$	82.9$_{\pm 0.9}$	85.9$_{\pm 0.1}$	87.6$_{\pm 0.1}$	66.3
	+STAGE	27.5$_{\pm 5.8}$	31.7$_{\pm 0.0}$	55.5$_{\pm 2.0}$	66.8$_{\pm 1.7}$	**86.6**$_{\pm 1.0}$	**86.1**$_{\pm 0.4}$	**87.8**$_{\pm 0.6}$	**89.9**$_{\pm 0.5}$	66.4
PC	RoBERTa	68.7$_{\pm 0.9}$	91.2$_{\pm 0.2}$	92.2$_{\pm 0.4}$	93.0$_{\pm 0.2}$	93.9$_{\pm 0.1}$	94.2$_{\pm 0.0}$	94.6$_{\pm 0.0}$	94.8$_{\pm 0.1}$	90.3
	+EDA	89.4$_{\pm 0.7}$	91.6$_{\pm 0.3}$	92.4$_{\pm 0.1}$	93.4$_{\pm 0.0}$	93.9$_{\pm 0.4}$	94.4$_{\pm 0.0}$	**95.0**$_{\pm 0.2}$	95.0$_{\pm 0.3}$	93.1
	+BT	89.5$_{\pm 0.2}$	90.9$_{\pm 0.5}$	92.3$_{\pm 0.0}$	93.4$_{\pm 0.0}$	93.8$_{\pm 0.2}$	94.4$_{\pm 0.1}$	94.6$_{\pm 0.1}$	94.9$_{\pm 0.0}$	00.0
	+AMR-DA	**90.0**$_{\pm 0.3}$	90.8$_{\pm 0.1}$	92.0$_{\pm 0.4}$	93.3$_{\pm 0.0}$	94.0$_{\pm 0.0}$	94.4$_{\pm 0.0}$	94.5$_{\pm 0.1}$	95.0$_{\pm 0.2}$	93.0
	+STAGE	89.5$_{\pm 0.7}$	**91.7**$_{\pm 0.3}$	**92.6**$_{\pm 0.3}$	**93.5**$_{\pm 0.1}$	**94.2**$_{\pm 0.2}$	**94.7**$_{\pm 0.1}$	**95.0**$_{\pm 0.1}$	**95.1**$_{\pm 0.1}$	**93.3**
IMDB	RoBERTa	90.4$_{\pm 0.5}$	91.2$_{\pm 0.4}$	92.2$_{\pm 0.2}$	**92.7**$_{\pm 0.3}$	93.6$_{\pm 0.1}$	93.4$_{\pm 0.1}$	**94.1**$_{\pm 0.2}$	94.1$_{\pm 0.2}$	92.7
	+EDA	90.3$_{\pm 0.4}$	90.8$_{\pm 0.2}$	91.8$_{\pm 0.1}$	92.2$_{\pm 0.1}$	93.3$_{\pm 0.1}$	**93.9**$_{\pm 0.0}$	94.0$_{\pm 0.0}$	94.0$_{\pm 0.1}$	92.5
	+BT	88.9$_{\pm 0.4}$	89.2$_{\pm 0.3}$	92.1$_{\pm 0.2}$	93.1$_{\pm 0.2}$	93.0$_{\pm 0.1}$	**93.9**$_{\pm 0.3}$	93.6$_{\pm 0.1}$	94.0$_{\pm 0.1}$	92.2
	+AMR-DA	90.1$_{\pm 0.3}$	90.5$_{\pm 0.2}$	90.9$_{\pm 1.3}$	92.4$_{\pm 0.3}$	93.2$_{\pm 0.2}$	93.2$_{\pm 0.3}$	93.2$_{\pm 0.6}$	93.8$_{\pm 0.2}$	89.8
	+STAGE	**91.4**$_{\pm 0.7}$	**91.5**$_{\pm 0.3}$	**92.3**$_{\pm 0.3}$	92.4$_{\pm 0.1}$	**93.8**$_{\pm 0.2}$	93.5$_{\pm 0.1}$	94.0$_{\pm 0.1}$	**94.2**$_{\pm 0.1}$	**92.9**

Table 21: **STAGE** performance on RoBERTa

τ	Data-ratio	1%	2%	5%	10%	30%	50%	70%	100%	avg
2	performance	79.5	83.3	87.4	88.5	89.4	90.0	90.9	91.3	87.5
	N_{aug}	300	500	1,747	3,499	9,999	17,497	24,995	34,596	-
5	performance	79.4	83.6	87.6	88.8	89.6	90.1	90.9	91.5	87.7
	N_{aug}	298	498	1,740	3,487	9,978	17,465	24,962	34,557	-
10	performance	**81.9**	**84.9**	**87.6**	**89.6**	**90.3**	90.7	**91.6**	**92.1**	**88.6**
	N_{aug}	294	493	1,727	3,476	9,957	17,440	24,922	34,511	-
20	performance	76.5	82.2	86.6	88.7	89.6	90.0	90.9	91.6	87.0
	N_{aug}	259	468	1,721	3,453	9,926	17,407	24,877	34,457	-
30	performance	72.2	82.0	87.1	88.5	89.5	**90.8**	91.4	92.0	86.7
	N_{aug}	183	421	1,693	3,446	9,982	17,367	24,846	34,422	-

Table 22: Performance comparison with various τ. N_{aug}: number of augmentation samples

Method		Training subset size ratio								Average
		1%	2%	5%	10%	30%	50%	70%	100%	
BERT		55.7	77.0	84.5	86.7	89.0	89.7	90.7	91.2	83.1
STAGE	DS	77.8	84.6	86.7	89.3	89.6	90.2	91.0	91.6	87.6
	DR	77.8	84.9	86.7	88.9	89.5	90.3	91.1	91.6	87.6
	DI	75.0	83.3	86.3	88.4	89.5	90.0	90.9	91.6	86.9
	SR	75.5	82.8	86.5	88.3	89.5	90.0	90.8	91.5	86.9
	SI	76.8	84.4	86.7	89.0	89.6	90.1	90.9	91.6	87.4
	RI	77.0	83.3	86.6	89.3	89.2	90.0	91.0	91.6	87.3
	DSR	78.5	84.1	87.3	89.0	89.9	90.5	91.3	91.6	87.8
	DSI	78.1	84.0	86.9	89.2	89.5	90.4	91.2	91.8	87.6
	DRI	77.4	84.8	87.1	89.1	89.6	90.5	90.9	91.6	87.7
	SRI	78.1	83.0	87.4	89.4	89.8	90.5	90.9	91.6	87.6
	DSRI	**81.9**	**84.9**	**87.6**	**89.6**	**90.3**	**90.7**	**91.6**	**92.1**	**88.6**

Table 23: Performance on various combinations of **STAGE**

G. Results with respect to τ

We have performed various ablation studies on τ, but we present the ablation results for four subsets in Section 6. We present the whole results on the 8 subsets of training data in Table 22.

H. Necessity of the Whole Operations

To verify the necessary of multiple operations, we partially apply each operation used in **STAGE**. The experimental results are presented in Table 23. In the table, ‘D’, ‘S’, ‘R’, and ‘I’ represent the chosen operations for **STAGE**, i.e., D-ME (delete), R-MC (replace), I-MEN (insert), and S-DSP (swap), respectively. If we combine only two operations, the average performance is 87.3%, and if we combine three operations, it is 87.7%. However, the whole combination is 88.6%. The experimental results demonstrate that the performance improvement is more significant if a wider range of operations are applied. This is attributed to the application of various modification methods, which results in obtaining diverse sentences. Through the experiments, the validity of our proposed combination of four operations, known as **STAGE**, is evident.