Communications for Statistical Applications and Methods
2023, Vol. 30, No. 3, 343–354

https://doi.org/10.29220/CSAM.2023.30.3.343
Print ISSN 2287-7843 / Online ISSN 2383-4757

Supervised text data augmentation method for deep neural networks

Jaehwan Seol[a], Jieun Jung[a], Yeonseok Choi[a], Yong-Seok Choi[1,a]

[a]Department of Statistics, Pusan National University, Korea

Abstract

Recently, there have been many improvements in general language models using architectures such as GPT-3 proposed by Brown *et al.* (2020). Nevertheless, training complex models can hardly be done if the number of data is very small. Data augmentation that addressed this problem was more than normal success in image data. Image augmentation technology significantly improves model performance without any additional data or architectural changes (Perez and Wang, 2017). However, applying this technique to textual data has many challenges because the noise to be added is veiled. Thus, we have developed a novel method for performing data augmentation on text data. We divide the data into signals with positive or negative meaning and noise without them, and then perform data augmentation using *k*-doc augmentation to randomly combine signals and noises from all data to generate new data.

Keywords: NLP, data augmentation

1. Introduction

Although there have been many improvements in the field of machine learning, technology and performance, especially with neural network models, it is still difficult to replace all text data-based neural network models because complex models require enormous computational resources. Many industries and researchers want to train their own text classification models on custom-prepared data, and the most serious problem researchers face is the lack of data. To address this problem, researchers usually modify models or process data. One of the most viable and effective technologies they use is data augmentation. The augmentation method has played an important role in improving model performance without collecting new data or modifying the model structure.

However, for text data, it is difficult or impossible to define the noise that needs to be added to the document. Therefore, we remove meaningless words and replace them with synonyms as opposed to the way image augmentation has performed in NLP tasks. However, workloads are still needed, and performance has not improved as successfully as image data augmentation. Because, it is difficult or impossible to define the noise to be added to the document. However, in a recent paper by Wei and Zou (2019), "EDA: Easy data scaling techniques to improve the performance of text classification tasks," researchers presented a language model and four simple text editing techniques (SR, RI, RS, RD) that can augment text without using external data and yields the same performance as when using 100% of the original data. We developed a novel data augmentation method that mitigates the data volume problem and improves model performance through user input, with the idea that there may be a more aggressive approach for text data augmentation. Section 2 introduces data, graphically

[1] Corresponding author: Department of Statistics, Pusan National University, 2 Busandaehak-ro, 63beon-gil, Geumjeong-Gu, Busan 46241, Korea. E-mail: yschoi@pusan.ac.kr

Published 31 May 2023 / journal homepage: http://csam.or.kr

Table 1: Reduced datasets from IMDB to Aug75

Data	Train	Test
IMDB	25,000	25,000
IMDB5736	2,868	2,868
Easy600	300	300
Aug75	75	300

describes a one-dimensional convolution neural network for data, and describes signal and noise. Section 3 introduces options and algorithms, and Section 4 presents experimental results based on these options and algorithms. Finally, a conclusion is drawn in Section 5.

2. Data and model description

IMDB data from Stanford University is used in the experiment. The data has 25,000 movie reviews for training and 25,000 movie reviews for testing. The number of reviews in a movie does not exceed 30, and the percentage of positive or negative reactions is the same for each movie, so if you guess a label randomly, the accuracy is 50%. To improve the data quality for simulation, the researchers classified reviews with a movie rating of 4 or less out of 10 as "negative" and those with 7 or more out of 10 as "positive". As a result, the dataset does not have a neutral review and is greatly polarized by both positive and negative emotions. This data consists of review text and review sentiment (Maas *et al.*, 2011). The review text is a converted word index or a one-hot encoding vector, and the review sentiment has category 0 for negative and category 1 for positive.

For a more detailed numerical representation of this data, x_{ij} is called an independent variable, an integer mapped from text to $0 \leq x_{ij} \leq v$. Here, v is the vocabulary size of the word, and is a response variable that is an integer of 0 (negative) or 1 (positive). Here, the number of data is the number of words of the i^{th} data, and j^{th} is the number of data.

Using 50,000 full data to show augmented effects is not appropriate for two reasons. First, the model trained with the entire dataset has sufficient performance above 0.9 accuracy, making it unlikely to be improved with other techniques. Secondly, large data requires more oversight. Thus, to show the effectiveness of this method more clearly, we establish several learning cases for simulations with very few samples while ensuring some performance. Therefore, we reduced the sample size in such a way that the sample does not lose too much information and results in a dataset with easier tasks, such as Katharopoulos and Fleuret (2018).

To this end, dimension reduction is performed on the observations on the data. For sentence classification, Kim (2014) proposed a one-dimensional convolution neural network (1D CNN) architecture that builds text classifiers using some datasets, including movie reviews. This model was very simple in both architecture and computational complexity. We chose this model to show that data augmentation is effective for the general and simple models used by researchers.

First, we trained the 1D CNN model with all samples less than 100 words long. IMDB5736 is a sample of 5736 in less than 100 words. Some samples are then selected to reduce the losses calculated in the model. In this way, we obtain an Easy600 dataset from the entire IMDB data. We use the Easy600 dataset to test different sample sizes to measure the impact of the different sample sizes and model performance baselines. And finally, the Aug75 dataset is collected as augmented source data for training. 300 data from Easy600 are re-used to test the model with augmented data.

Table 1 lists the data sets obtained in this section. IMDB is the original data set. IMDB5736 and Easy600 are reduced datasets in R-technology. We will use Aug75 as the source data for data augmentation.

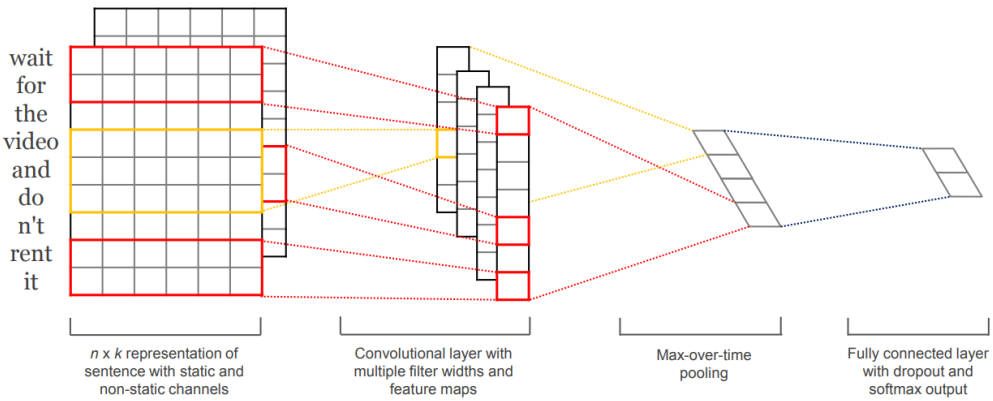

Figure 1: *1dConv layer: One-Dimensional convolutional neural networks for sentence classification.*

Figure 1 is a general description of 1D CNN by Kim (2014). This figure shows the convolution process applied to one-dimensional text data. This sentence, "Wait for a video and don't rent it," is converted into a matrix in the form of the next sentence after going through a layer of tokenization, padding and embedding. In the Figure 1, n is the length of the sentence and k is the dimension of the embedding vector. We obtain the result vectors in two kernels with sizes 2 and 3, respectively, and show the maximum time pooling the largest value in each vector. The neural network to be designed is a neural network for binary classification with two neurons in the output layer. All scalar values obtained by this method are connected and made into vectors. Text classification is performed by fully connecting two neurons to the output layer.

Figures 2 and 3 show the model configuration. Figure 2 illustrates how text data is treated as input vectors in the convolution layer by describing the sequential process of creating word representations. Create vector values for 1000 words to use for one-hot encoding. Then, apply one-hot encoding to the sentence "This movie is very good". Zero padding was performed as long as the length of the sentence to prevent data reduction, and word embedding was performed and express in dense vector form. Thus, we obtain embedding vectors, which are low-dimensional real matrices. The sequential information flow shown in Figure 3 is a neural network model that begins after the embedding process described in Figure 2. As shown in Figure 3, the 1D CNN model has an embedding layer that uses an embedding weight matrix without prior training. First, the embedded word vector generated in Figure 2 undergoes a Conv1D, i.e., a one-dimensional convolution process, and applies the activation function ReLU to take the negative value to zero. We repeat this one more time and then extract the maximum of certain values within the kernel via the max pooling process. Unlike the first and second processes, additional dropouts from the third process are applied randomly to remove some neurons in the learning process to prevent overfitting. In the dense process, the resulting values are linked using matrix multiplication. Then, apply dropout and ReLU twice in succession. Finally, the binary cross-entropy function is obtained by applying the density process and the activation function sigmoid function.

The loss function is a binary cross-entropy function, which is common in binary classification with a formula of "loss $= -\sum_{i=1}^{n} y_i \times \log(\hat{y}_i)$". Here, y is a discrete value of 0 or 1, and $\hat{y}$ is the output value of the sigmoid function. The Adam optimization tool is used as an optimization tool.

To ensure that the data quality fits the model, the data was processed into a signal noise structure

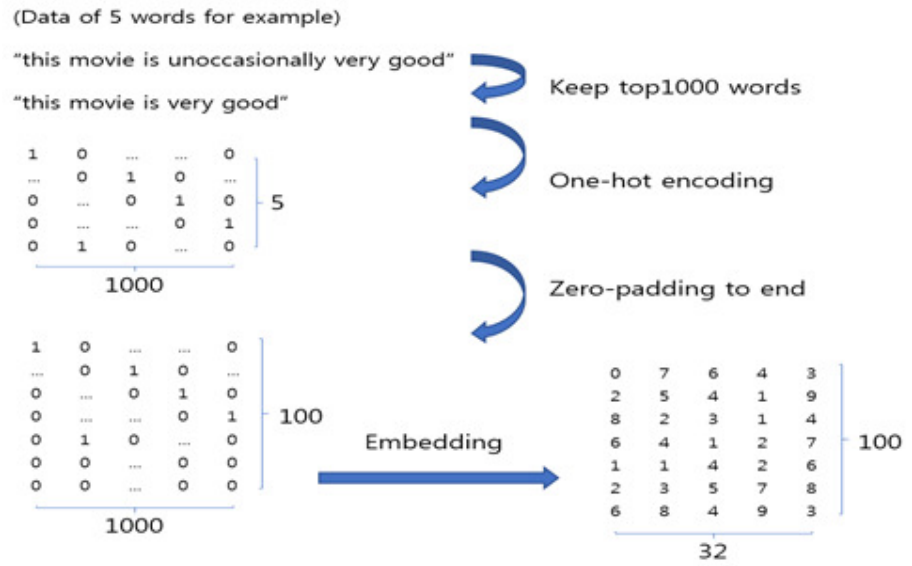

Figure 2: *Process to get embedded word vectors.*

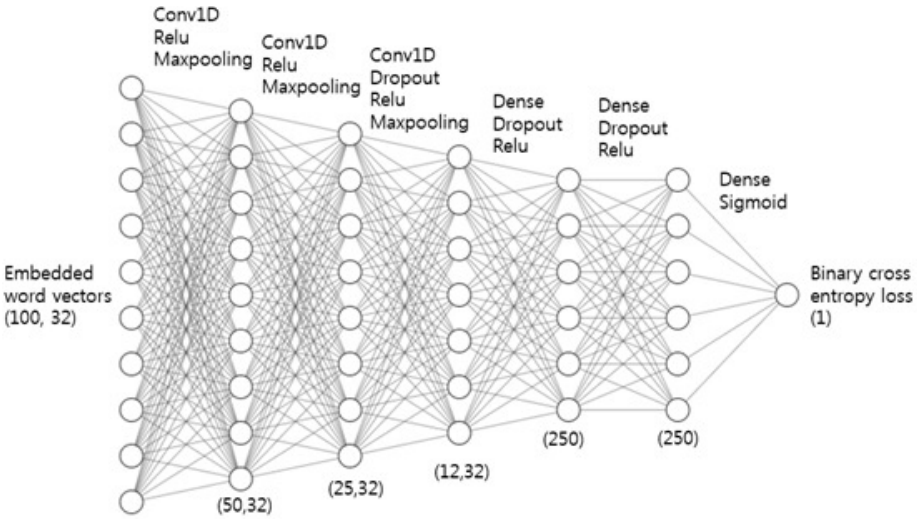

Figure 3: *1D CNN diagram.*

before enlargement (Lehrer, 2017). A "signal or key feature" means a variable that is decisive and consistent with the response variable, while "noise or error" means an unwanted occasional random turbulence. Applying the proposed augmentation method requires metadata from the original data set indicating which part of the data is the signal and which part is the noise. Therefore, for this key function (i.e., signal) selection process, we created a custom labeling tool that can easily emphasize signals and exclude noise. Supervision is performed by using the tools in this document. The user can quickly check the signal part. And expressions other than signals are considered noise.

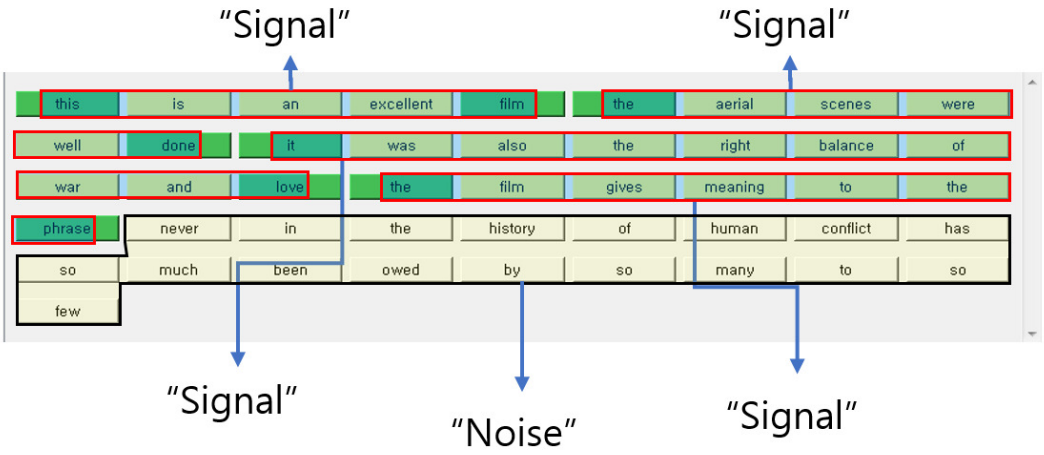

Figure 4: *Figure of labeled signals and noises.*

Table 2: Labeling result summary

Meta label	Sequence	Word token	Unique token
Negative signal	174	1,825	708
Positive signal	145	1,647	658
Neutral noise	108	1,137	539

Figure 4 is a custom labeling tool that can easily emphasize signals and separate noises. You can assign a series of text to a 'signal' by directly clicking on two words. At first, when a user clicks on a word in a document, the color of the word changes to dark gray. And when you click on another word, the series of words between the two clicked words are highlighted with a light gray signal. Clicking the same button again cancels the selection of the area that indicates the area is no longer a signal. All other words not in the signal part are considered noise in the later process. The focus is on signals, which are direct expressions which give the impression that the review is positive or negative. Signals are direct suggestions and the rest is noise.

'Sequence' is a word classified as signal noise, and 'word token' is the number of tokenized words among words. Also, 'unique token' is the number of tokenized words that mean 'negative', 'positive', and 'neutral' in signal noise.

As a result, there were 174 words in the signal pool of negative reviews, 145 words in the signal pool of positive reviews, and 108 words in the noise pool as shown in Table 2.

3. Data augmentation

The basic principle of data augmentation is to match the signal to the label while adding random noise to the data. After signal noise labeling, we have text data and meta-label data that clarify which text is signal or noise.

This data augmentation method requires user input. The input is either a signal function of a category or a noise function of all categories. Based on 'signal noise theory', we create different combinations of signal and noise (Kay, 1993).

Table 3 describes important options for understanding further modifications and applications of

Table 3: List of important options for data augmentation

Option name	Usage	Range	Used
Sequence-Wise features	Whether a pool consists of each token or a sequence of token	Ture, False	True
Signal-Noise ratio	Probability of more signals than noises	0 ~ 1	0.6
Target_length	Maximum, proper size of length for each augmented data	Int, None	300
Minimum_length	Minimum length	Int, None	100
Target_length_policy	Length distribution for each augmented data	Gaussian, constant	Gaussian
Swell-Ratio	Size of augmentation. Times of original data	Number	200, 500
Ndocs	How many documents are referred to make one data	Dictionary	{1:0.2, 5:0.3, n:0.5}
Shuffle_pools	Remove order effect when referring data from ndocs	True, False	True
Stable_pickup	Pick in a way less-chaotic. No picking the same signal or noise is allowed unless the other is all used up.	True, False	True
Original_ratio	Original training data ratio to augmented dataset to be created	Number	0

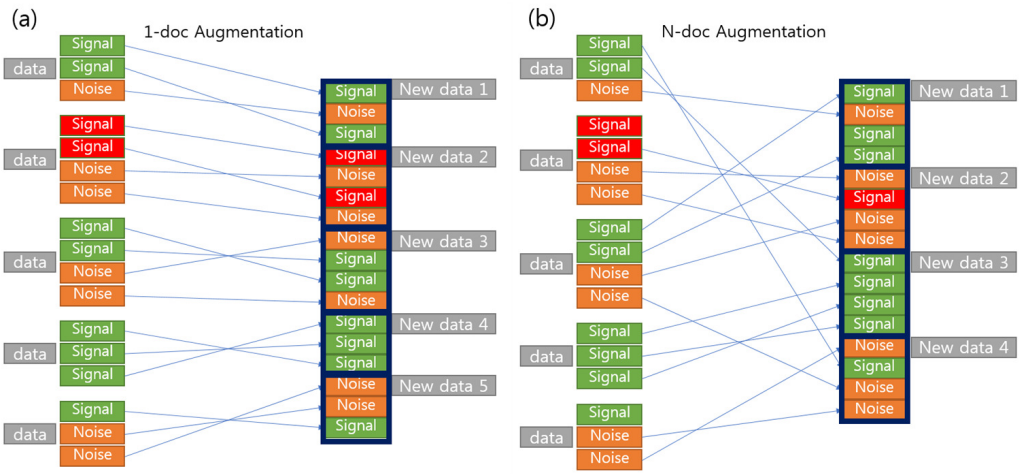

Figure 5: *(a) 1-doc augmentation, (b) n-doc augmentation.*

augmented processes.

One of them, Sequence-wise features, is true, and the signal pool and noise pool are labeled with word sequences. If this option is False, all statements of signal or noise will be distributed into words. In other words, N docs is k-doc. K-doc refers to the enlargement process for creating new data for one of the k documents. For example, when the magnification described in this chapter is performed on all samples (i.e., $k = N$), the noise and signal can vary widely, which is the goal of the magnification method. In 1-doc augmentation, one augmented data consists of only one signal and noise of the original data, while n-doc augmentation refers to the signal and noise pool of all data. A dictionary is used to specify a ratio for each k-doc amount.

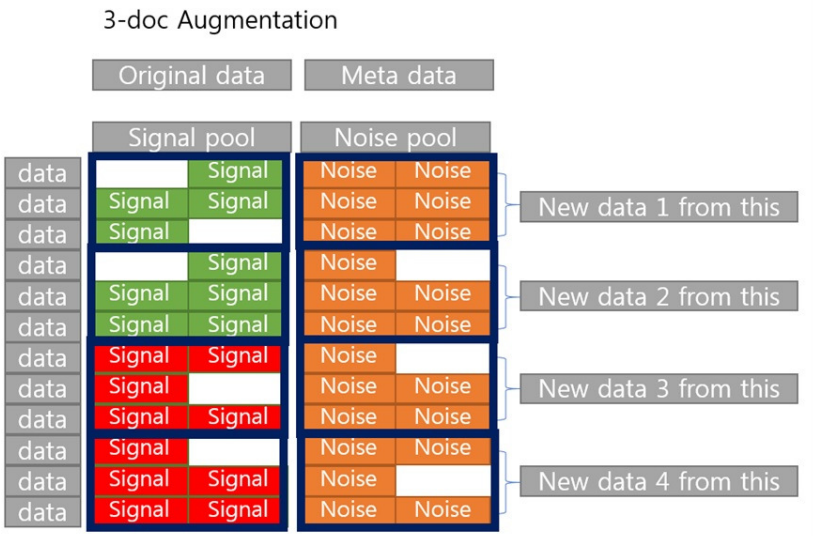

Figure 6: *1dConv layer: One-Dimensional convolutional neural networks for sentence classification.*

Figure 5 shows how k-doc augmentation works in terms of signal and noise. The gray signal is a positive signal = 1, and the black signal is a negative signal = 0. Noise is not a categorical difference. In Figure 5(a) each new data made with 1-doc augmentation comes from only one of the original data, whereas Figure 5(b) has a signal and noise pool for all the original data with n-doc augmentation.

Since the idea that such information could have miraculous potential in data augmentation, we decided to actively create a fairly dangerous and entirely new combination of signals and noises from scratch instead of carefully adjusting each sample from the original data.

For understanding, Figure 6 shows how signal pools are generated for noise pools of k-doc, yi, and k-doc.

Similar to the processes in Figure 5(a) and Figure 5(b), we describe how signal pools and noise pools are constructed in different views.

The following is an example of a data augmentation process. When the data augmentation option is set to Table 3, the data augmentation process operates as follows.

Step 1 : Make loop for each parameter

For each k-doc in [1, 5, n]

For each y in [0, 1]

Do [Step 2] ~ [Step 5].

Step 2 : Calculate swell_size

swell-size = k-doc rate $*$ # of data with y_i $*$ swell-ratio

Repeat [Step 3] ~ [Step 5] while augmentation count < swell-size.

Step 3 : Generate random value of target_length

target_length ~ Normal(300, (10 + 300/4)).

Step 4 : Repeat below while new_target < target_length

Decide either signal or noise based on signal-noise ratio.

If signal, add one sequence of text from signal pool of k-doc, y_i.

If noise, add one sequence of text from the noise pool of k-doc.

If stable pickup is true, selection is without replacement and therefore, if a pool is empty, refill it.

If stable pickup is false, selection is with replacement.

Step 5 : Minimum length check

If the length of created data is below minimum_length, discard the data.

Step 6 : Loop end

When augmentation is done for each k-doc, y_i, merge all k docs. Add original data with respect to the original ratio. Return accumulated data as an augmented data set.

[Positive review generated from augmentation]

You'll love them its been a few years since i saw it and nothing has come close since then and in this movie shaq plays a genie who lives in a boom box is that not original a genie in a boom box instead of a lamp beautifully constructed tells the story of narcotics usage and commerce from multiple points of view.

Table 4 shows the relationship between the original data and the augmented data generated by the augmentation. The signal of positive reviews and the noise of negative reviews are combined to construct augmented data. In Table 4, we can check the context of sentence combinations and how noise and signal affect the category orientation of the document.

At first glance, the entire sentence appears to be positive because there are positive signals in the review in Table 4. However, almost half of the sentences come from negative reviews. This impression explains that the overall direction of the review is actually drawn by the signals in it. In other samples, the subject is sometimes out of context but generally suitable for categories.

4. Experiment and result

The 300, 200, 125, 75, 40, 25, and 10 training data from the Easy600 and 300 test data from the Easy600 are used to experiment with performance changes at various sample sizes. 75 training data from Aug75 and 300 test data from Easy600 are used when experimenting with performance improvement in augmentation. Aug75 was collected by random sampling from Easy600 with seed number 11. In addition, the model used was 1D CNN. Table 5 shows the detailed architecture of the model. Three convolution layers and two density layers are mainly used. The dropout normalization method is simply added, and the last output is sigmoid, also known as a logistic function.

Both the original data and the augmented data are balanced by the accuracy (i.e., verification accuracy) calculated from the test data set. The performance metric for models with sample size N is defined as the average accuracy of fit points among various samples with size N. The fit point is the iteration point of the parameter update, the number of epochs between underfitting and overfitting, which reliably maximizes verification accuracy.

Figure 7 illustrates how fit points are determined to evaluate model performance. Thus, the fitting point is a point within the interval between the 10^{th} epoch and the 14^{th} epoch. In this section, the

Table 4: Composition of augmented data from the view of original data

Data source	Augmented data
Positive signal from 16	you'll love them
Noise from 17	its been a few years since i saw it and nothing has come close since then and
Noise from 5	in this movie shaq plays a genie who lives in a boom box is that not original a genie in a boom box instead of a lamp
Positive signal from 18	beautifully constructed tells the story of narcotics usage and commerce from multiple points of view

Table 5: Model summary

Layer	Output shape	Number of parameters
Embedding	(100, 32)	2,825,184
Conv1D	(100, 32)	3,104
Conv1D	(50, 32)	3,104
Conv1D	(25, 32)	3,104
Dense	(250)	96,250
Dense	(250)	62,750
Dropout	(250)	0
Activation	(1)	0

accuracy converges to 0.800, and maximizes at approximately 0.810. The appropriate convergence of model performance is evident.

Each line in Figure 8 is a graph of 30 different seeds with a fixed sample size N. The increased accuracy drawn on the graph shows how well the model can learn from a limited number of N data. For all 300 training data from Easy600, the accuracy increased by up to 0.820 on the mean of different samples. However, 10 randomly sampled training data from Easy600 showed a limited accuracy increase of around 0.530, close to the baseline accuracy of 0.500 for binary classification using random guesses. The reference performance to be compared with the model performance after enlargement is 0.64 for reference performance considered in the conditional sample size is 75; and it is 0.66 for only for the reference performance of Aug75.

In the experiment, all augmentation options are tested individually. In these tests, most of the augmentation parameters did not affect data quality. Since only the signal noise ratio and swelling ratio options were different, it only affected the augmented quality. Augmented results were good when the signal noise ratio was 0.600 and the swelling ratio was large enough. Therefore, there was performance improvement according to Table 3.

After reinforcement with Aug75 training data, we obtained approximately 11,500 new data for the swelling ratio 200 and 29,000 data for the swelling ratio 500. For each seed from 0 to 29, we used these augmented data to train a 1D CNN model.

In the top graph of Figure 9, Aug_swell-ratio500 is the average graph of seed numbers 0 to 29. It is trained with new data augmented on Aug75 data with ratio500. The Aug_swell-ratio200 graph shows similar results with a different swell-ratio of 200. The Aug75 graph is a single graph of 1D CNN trained with a Aug75 data. The Size75_average graph is an average graph of seed numbers 0 to 29 trained with 75 different samples from 300 training data from Easy600.

In Figure 9 and Table 6, we consider conformity to be epoch 30. Therefore, as a result of learning with about 29,000 data generated through 500 augmentations, the average performance increased by

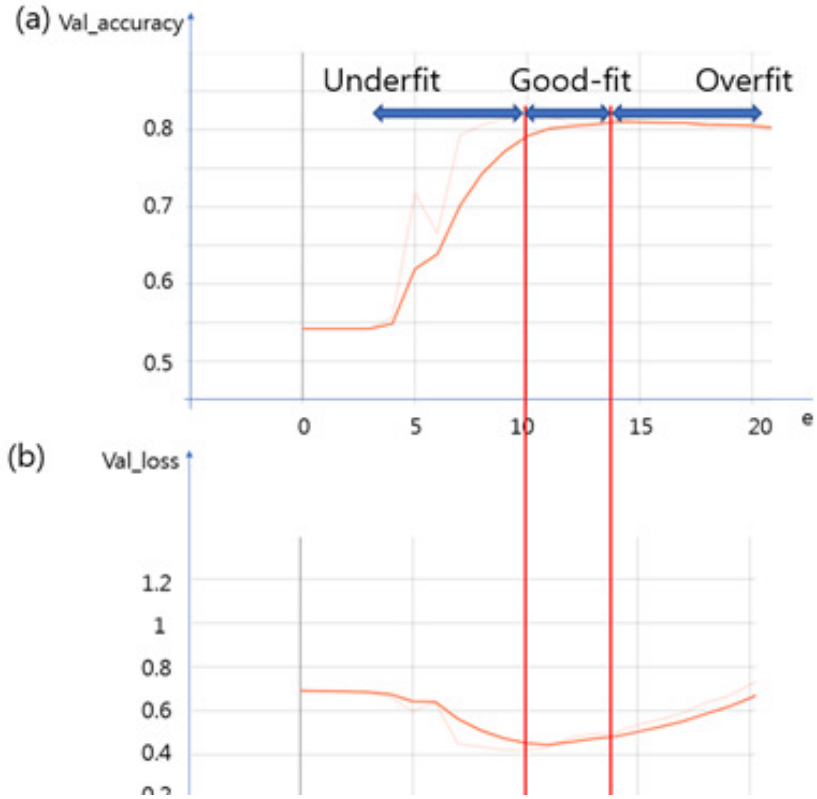

Figure 7: *Illustration of good-fit points based on two graphs of the same training (a) graph of validation accuracy (b) graph of validation loss.*

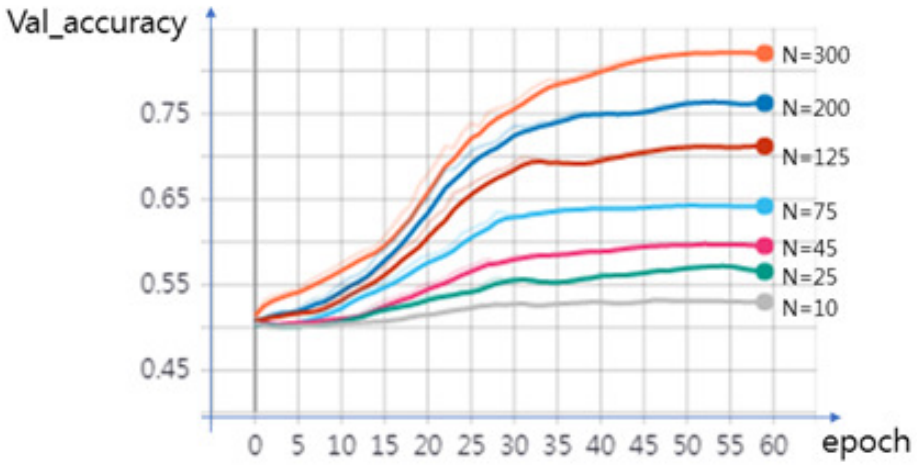

Figure 8: *Performances per sample sizes N(300, 200, 125, 75, 40, 25, 10) of Easy600.*

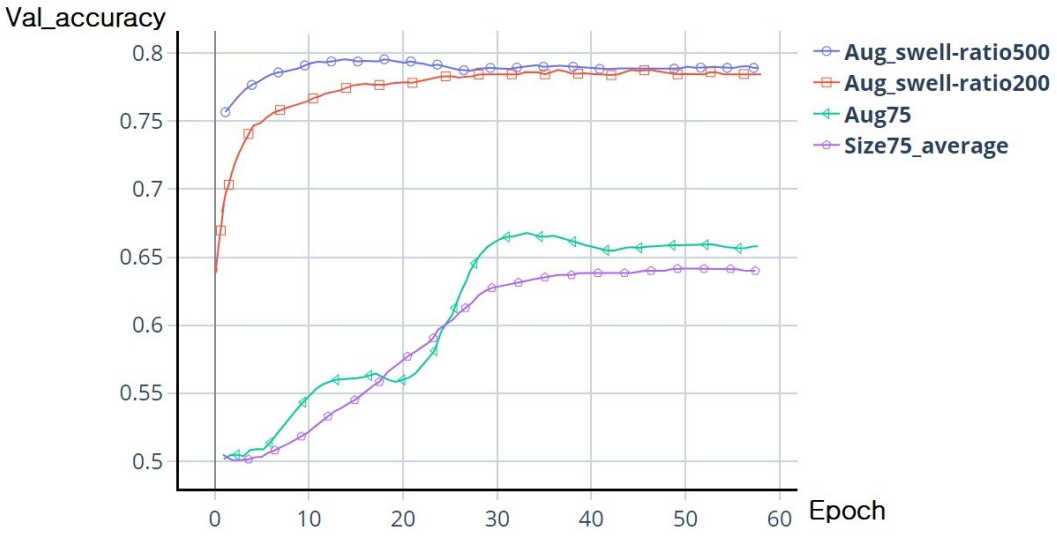

Figure 9: *Graph performance comparison.*

Table 6: Performance comparison with metrics at each epoch

Name	Accuracy		
	Epoch 10	Epoch 30	Epoch 50
Aug_swell-ratio500	0.793	0.789	0.790
Aug_swell-ratio200	0.765	0.785	0.785
Aug75	0.549	0.663	0.659
Size75_average	0.521	0.628	0.642

about 0.161 in Figure 9 compared to the case of learning with 75 data in Table 6.

The model trained with the original data without augmentation made the maximum accuracy convergence 0.628 in an average of 30 trainings of different samplings. However, the model trained with augmented data without original data made the maximum accuracy convergence 0.789 in an average of 30 trainings of different random augmentations when the augmentation is 500 times that of 75 samples.

5. Conclusion

At baseline performance of accuracy 0.5, the original training datasets showed 0.7 accuracy. Training datasets created through text data augmentation helped the model improve up to 0.8. Thus, if researchers attempt other augmentation methods or feature engineering methods, such as synonym replacement or word weight, our method has the advantage of exploiting almost all the potential of metadata used for augmentation to improve model performance. The data augmentation effect is expected to be the same as collecting approximately three times the original data because the performance of the Aug75 data after augmentation was above accuracy when 200 training data were used for learning. At supervision cost, this augmentation method is particularly useful when creating a simple, less computational model for the few-shot learning case (Snell *et al.*, 2017). If supervision is relatively burdensome, this problem can be addressed by applying different model-based signal

locations, whether pre-trained or not.

Using this data augmentation method as part of the model architecture allows us to build an effective model that addresses the signal and noise structures of text data. We hope that this data augmentation method will be used to break through a small amount of data problems in the current NLP field.

References

Brown TB, Mann B, Ryder N *et al.* (2020). Language models are few-shot learners, Available from: *arXiv preprint arXiv:2005.14165*

Katharopoulos A and Fleuret F (2018). Not all samples are created equal: Deep learning with importance sampling, Available from: *arXiv preprint arXiv:1803.00942*

Kay SM (1993). Fundamentals of Statistical Signal Processing, *Prentice Hall PTR*, NewJersy.

Kim Y (2014). Convolutional neural networks for sentence classification, Available from: *arXiv preprint arXiv:1408.5882*

Lehrer R (2017). Modeling signal-noise processes supports student construction of a hierarchical image of sample, *Statistics Education Research Journal*, **16**, 64–85.

Maas A, Daly RE, Pham PT, Huang D, Ng AY, and Potts C (2011). Learning word vectors for sentiment analysis, In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, Portland, Oregon, USA, 142–150.

Perez L and Wang J (2017). The effectiveness of data augmentation in image classification using deep learning, Available from: *arXiv preprint arXiv:1712.04621*

Snell J, Swersky K, and Zemel R (2017). Prototypical networks for few-shot learning, Available from: *arXiv preprint arXiv:1703.05175*

Wei J and Zou K (2019). Eda: Easy data augmentation techniques for boosting performance on text classification tasks, Available from: *arXiv preprint arXiv:1901.11196*

Received November 04, 2022; Revised January 11, 2023; Accepted January 26, 2023