symmetry

MDPI

Article

Self-Supervised Contextual Data Augmentation for Natural Language Processing

Dongju Park and Chang Wook Ahn *

Electrical Engineering and Computer Science, Gwangju Institute of Science and Technology, Gwangju 61005, Korea; toriving@gist.ac.kr

* Correspondence: cwan@gist.ac.kr; Tel.: +82-62-715-2661

Received: 10 October 2019; Accepted: 7 November 2019; Published: 11 November 2019

Abstract: In this paper, we propose a novel data augmentation method with respect to the target context of the data via self-supervised learning. Instead of looking for the exact synonyms of masked words, the proposed method finds words that can replace the original words considering the context. For self-supervised learning, we can employ the masked language model (MLM), which masks a specific word within a sentence and obtains the original word. The MLM learns the context of a sentence through asymmetrical inputs and outputs. However, without using the existing MLM, we propose a label-masked language model (LMLM) that can include label information for the mask tokens used in the MLM to effectively use the MLM in data with label information. The augmentation method performs self-supervised learning using LMLM and then implements data augmentation through the trained model. We demonstrate that our proposed method improves the classification accuracy of recurrent neural networks and convolutional neural network-based classifiers through several experiments for text classification benchmark datasets, including the Stanford Sentiment Treebank-5 (SST5), the Stanford Sentiment Treebank-2 (SST2), the subjectivity (Subj), the Multi-Perspective Question Answering (MPQA), the Movie Reviews (MR), and the Text Retrieval Conference (TREC) datasets. In addition, since the proposed method does not use external data, it can eliminate the time spent collecting external data, or pre-training using external data.

Keywords: data augmentation; self-supervised learning; natural language processing; text classification

1. Introduction

The rapid development of effective and efficient machine learning and deep learning has changed the paradigm of methodologies in various fields. In particular, the neural network-based model provides exceptional performance in a variety of computer vision (CV) tasks including image classification [1], image generation [2], semantic segmentation [3], and object detection [4]. It also performs well in natural language processing (NLP), such as machine translation [5], language modelling [6], question answering [7], sentiment analysis [8], and text classification [7].

Recently, various variants of convolutional neural networks (CNNs) [9], recurrent neural networks (RNNs) [10], and Transformer [11] model structures are used to improve performance. Moreover, training techniques for neural networks, such as label smoothing [12], learning rate decay [13], and transfer learning [14], have been used to improve learning efficiency or performance. In addition to these studies, the automatic hyperparameter optimization method for data and models, automated feature learning for extracting only necessary features from input data, and neural architecture search for constructing a suitable model for data, are also being studied [15,16].

To achieve good performance on the neural network models, both the structure and the data used on the training process are important. The amount and quality of data affect the performance of most machine learning models. However, the size of the data most neural models take in as input

often turns out to be unsatisfactory. Such deficiencies can lead to incomplete training or overfitting that causes failure in generalization. Solving the problem of neural network overfitting is one of the important areas of research in deep learning. Among the various studies, dropout [17] and batch normalization [18] are known as ways to overcome overfitting well. However, even these methods are not effective when the amount of data is small.

In order to alleviate the aforementioned problem, many studies take advantage of augmentation of the existing data [9,19–24]. Data augmentation is used for small amounts of data, but also unbalanced data. In addition, data augmentation has become almost indispensable in improving the performance of CV application and has been followed by various studies on other applications as well. In the CV field, image data can be easily augmented using various methods, such as cropping, rotation, scaling, shifting, and noise addition. Beyond the technique of augmenting the data similar to the original image, methods for augmenting the image with a large difference from the original image are being studied, as in Cutout [25], Mixup [26], and CutMix [27].

On the other hand, in the NLP field, data augmentation methods are not so popular, as text data are discrete and cannot be completely transformed into a continuous space, so the methods above cannot be used. For this reason, in general, NLP uses a method of replacing a specific word with a synonym from a thesaurus. Another method is to replace any word in the sentence with the closest word in the embedding space [21]. However, this method can be applied only when the words share the same or similar meanings in context, otherwise, it is difficult to expect good performance. Furthermore, the method of data augmentation using only words in a sentence does not reflect the context well. In order to solve this problem, contextual data augmentation methods have been studied [23,28,29].

In this paper, we propose a new data augmentation method that finds replacements for particular words under a certain context. Instead of the shallow concatenation of an independently trained forward and backward language model (LM), it uses deep bidirectional LM, in order to better understand contextual information. Unlike shallow bidirectional LM, which concatenates two LMs such as forward LM and backward LM, deep bidirectional LM can recognize bidirectional representation with a single LM. A deep bidirectional LM can be designed using a masked language model (MLM) [8], a self-supervised model of randomly masking words in a sentence and then finding the original words from the surrounding context. Also, we propose the label-masked language model (LMLM), which has been enhanced to effectively employ the MLM in data with annotation information. In the existing MLM, only one mask token was used, but the proposed LMLM has mask tokens for each label class. In addition, the proposed method conducts self-supervised learning upon the LMLM with only the given data of the task rather than fine-tuning the task-specific dataset after pre-training the external data in advance. In short, our method allows the original dataset to generate additional data.

We demonstrate that the proposed method boosts the classification accuracy of RNN [10] and CNN [30] based classifiers through various experiments for text classification benchmark datasets, including the Stanford Sentiment Treebank-5 (SST5), the Stanford Sentiment Treebank-2 (SST2), the subjectivity (Subj), the Multi-Perspective Question Answering (MPQA), the Movie Reviews (MR), and the Text Retrieval Conference (TREC) datasets. The proposed method shows an increase of 0.5–2.6% depending on the type of dataset and neural classification, compared to the model without data augmentation. Also, compared with the method proposed by Kobayashi (baseline) [29], the RNN model has improved the classification performance in all datasets, while the CNN model is improved only in some datasets. However, Kobayashi's method is time-consuming because it requires pre-training using external data. As our proposed method does not pre-train, it takes less time than Kobayashi's method. Kobayashi's method took about 20 h to pre-train and fine-tune, but our proposed method took less than two hours for each dataset.

The remainder of the paper is organized as follows: In Section 2, we review the related works about self-supervised learning, data augmentation, Transformer, and masked language model. Section 3 introduces the proposed method and the proposed model. The experiment, dataset, baselines,

hyperparameters, and results are discussed in Section 4. Finally, in Section 5, we present the study's conclusion, including discussion and future works.

2. Related Work

2.1. Self-Supervised Learning

The self-supervised learning method is one of the unsupervised learning approaches. This method automatically creates a labeled dataset using a given dataset and uses the generated dataset to help the model learn feature representations. It has been used extensively in various domains such as reinforcement learning [31–33], CV [34–36], and NLP [8,37]. Self-supervised learning can be as simple as predicting how rotated an image is [35] or solving a puzzle made from an image [36]. Another task is restoring the erased part of an image [38]. Recently, in the NLP field, deep bidirectional LMs, such as MLM, are trained through fill-in-the-blank tasks, which finds the original token from the masked token in a given sentence [8].

Normally, self-supervised learning is used as an auxiliary training process as well as pre-training before the downstream tasks training process. Unlike as mentioned above, we do not use self-supervised learning for pre-training or auxiliary training. On the contrary, in our proposed method, the self-supervised learning is employed to train the MLM for a regular training phase.

2.2. Data Augmentation

Common data augmentation methods for image data are rotation, cropping, scaling, shifting, and noise addition. Krizhevsky et al. [9] adjusted RGB channels, transformed the image, and made horizontal reflections to reduce overfitting. Zhong et al. [19] performed data augmentation using a method of erasing part of an image and filling the erased part with random values. Similarly, DeVries and Taylor [25] removed certain parts, but proposed a method to fill the erased parts with zeros instead of random values. In addition, new data can be created by applying weighted linear interpolation to data and labels, respectively [26]. Instead of erasing parts of the image and filling it with zeros, Yun et al. [27] used the method of filling the erased regions with parts of another image. The labels are also blended according to the proportions of the combined images. This method has achieved state-of-the-art performance on CIFAR and ImageNet classification tasks.

The text data augmentation method has been studied for various NLP tasks. Zhang et al. [20] arbitrarily selected a word in the sentence and replaced the word with synonyms from multiple thesauruses. Wang and Yang [21] have proposed replacing words in sentences with their neighbors in embedding space. Kafle et al. [22] used task-specific heuristic rules and long short-term memory (LSTM)-RNN [39] LMs to create new sentences. Fadaee et al. [40] have used bidirectional LSTM-LM for low-resource neural machine translation and to create new data by replacing words in sentences with rare words. Gao et al. [28] used soft contextual data augmentation for neural machine translation. The method is replacing randomly chosen words in a sentence with a mixture of multiple related words based on a distributional representation. The contextual data augmentation studies [23,29], which are the most similar to ours, used a bidirectional LSTM-LM or MLM to change the words in a sentence using a fill-in-the-blank task. These contextual data augmentation methods achieved the state-of-the-art results on the text classification benchmark dataset.

The audio data augmentation methods have been proposed to perturb the speed of the data [41], add noise to the data [42], and drop out the features of the data [43]. Park et al. [44] have achieved state-of-the-art performance by erasing data in a specific frequency region or erasing data at a specific time region.

2.3. Transformer and Transformer Encoder

The Transformer method [11] consists of transformer encoders and transformer decoders. The transformer encoder and decoder are based on self-attention rather than recurrence and

convolutional layers. The self-attention, also known as intra-attention, computes a weighted sum of the features at all tokens by attending to all tokens within the same sentence. In the transformer encoder, self-attention is calculated as multi-head attention that includes scaled dot-product attention. When the input is given the value Q for queries, K for keys, V for values, and d_k, the dimension of keys, scaled dot-product attention is calculated as follows [11]:

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V.$$

By performing scaled dot-product attention h times, the assorted attention information can be obtained. This approach is called multi-head attention. Given the projection parameters W_i^Q, W_i^K, W_i^V and W^O, multi-head attention can be computed as [11]:

$$\begin{aligned} \text{MultiHead}(Q, K, V) &= \text{Concat}(\text{head}_1, \ldots, \text{head}_\text{h})\, W^O \\ \textbf{where } \text{head}_\text{i} &= \text{Attention}(QW_i^Q, KW_i^K, VW_i^V). \end{aligned}$$

Then, we performed two linear transformations using the values obtained from the multi-head attention. The Rectified Linear Unit (ReLU) activation function is used between linear transformations. Both W_i and b_i are learnable parameters and have a dimension of d_{ff}. This operation is called a position-wise feed-forward network [11].

$$\text{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2.$$

In general, in the transformer encoder, Q, K, and V are the same values, indicating the input of the Transformer. The input of the Transformer consists of not only the token embedding, but also the positional encoding that injects the relative or absolute position of the token to handle the sequential data. The transformer encoder has the advantages of the parallelization found in a CNN and the sequential characteristics of a RNN, and is used for various NLP tasks with long-term dependency.

2.4. Masked Language Model

Bidirectional encoder representations from transformers (BERT) [8] has recently been used in most areas of natural language processing and has achieved state-of-the-art results. To train a deep bidirectional LM, BERT proposed the masked language model by referring to the Cloze Task [45]. The MLM, which directly implements bidirectional LM using the Transformer encoder including self-attention layers, token embedding, segment embedding, and position embedding, achieves better performance than a shallow bidirectional LM, which is indirectly implemented by concatenating left-to-right and right-to-left LMs. The MLM randomly masks input words and then predicts the original words only based on their context. Using MLM to predict the original word from a masked token is called an MLM task. Furthermore, it was used to pre-train a large quantity of external text data before task-specific fine-tuning in the BERT.

3. Proposed Method

We developed our method based on the intuition that the contextual representation of data depends on the domain of the dataset. For example, since the SST5 [46] dataset is in the domain of movie reviews, there is little data or terminology about politics. Therefore, there is no increase in performance when the pre-trained data does not share a similar domain with the data to be augmented. Although pre-training can be performed with a large amount of data for various domains, it creates a large number of parameters and a long training time. Accordingly, we train the proposed model, the label-masked language model, through self-supervision on given data without pre-training on external data. In other words, our proposed method does not pre-train, which contributes to reducing the time spent on training.

In summary, the proposed method uses deep bidirectional LM (label-masked language model) to increase the contextual representation of the model inferring masked words through the surrounding context compared to shallow bidirectional LM. Additionally, it eliminates the need for external data and reduces the procedure for data augmentation.

3.1. Label-Masked Language Model

The goal of the MLM, deep bidirectional LM, is to predict the masked word using the surrounding context after randomly masking words in the given sentence. In the proposed method, data augmentation is performed with previously used data, without pre-training. Therefore, prediction and learning are performed only on masked words.

We propose the label-masked language model (LMLM), which has been enhanced to effectively employ the MLM in data with label information. The LMLM has the same input and output sentences, but some words are masked at the input, making the input and output asymmetric. This asymmetry and the labeled mask token allow the model to learn the relationship between masked words and a surrounding context. In the existing MLM, only one mask token was used, but the proposed LMLM employs labeled mask tokens for each label class. Label information is important when performing the MLM task on labeled data. We mask words using mask tokens that represent the labels of a given sentence. Then LMLM learns to find out what words the labeled mask tokens originally are. This learning process allows LMLM to infer words based on the label information of the mask token. It also helps to ensure that newly created sentences with data augmentation do not have labels that differ from the original sentences. If the mask token does not contain label information, the masked token can be replaced by a word with a different label. For example, if we put a mask token that does not include label information for the "fantastic" token in the positive sentence "The actors are fantastic", it can be replaced by a negative word such as "boring" or "bad" by the surrounding context. Also, this situation can lead to a decrease in the performance of the classifier. Thus, we defined the label of the mask token by applying the label of the given sentence. In the dataset, the label of each sentence is pre-defined (the labels depend on the dataset. For instance, it can be positive or negative in a binary classification task, or it can be a named entity in a named entity recognition task), and we specified a word mask based on the label of the given sentence. For example, if the label of a given sentence is positive, a positive mask is automatically applied to every randomly selected word in the given sentence. In other words, we replace the selected word token with a positive mask token. The positive mask is an indication that a given mask token comes from a positive sentence. We created a labeled mask token for each label based on the dataset. Figure 1 shows the top five words predicted by the trained LMLM from the mask token along with the label information for each word.

Unlike the original MLM in BERT, the LMLM does not use segment embedding. The LMLM consists of position embedding, token embedding, and only one transformer encoder layer. Furthermore, when a word to be masked is selected, it is replaced by a masking token unconditionally without replacing with the other word or retaining. Although LMLM—the model we use in our proposed method—has fewer parameters than the originally proposed BERT's MLM, the number of parameters is sufficient to build a structure for the proposed method while maintaining a rapid learning speed. The overall structure of LMLM is shown in Figure 2.

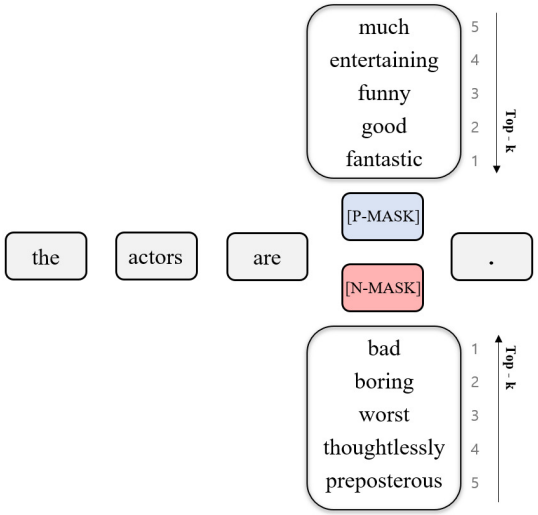

Figure 1. Top five words predicted by the model from the two different labeled mask tokens for the sentence "the actors are [MASK]." [P-MASK] is the positive mask token, and [N-MASK] is the negative mask token.

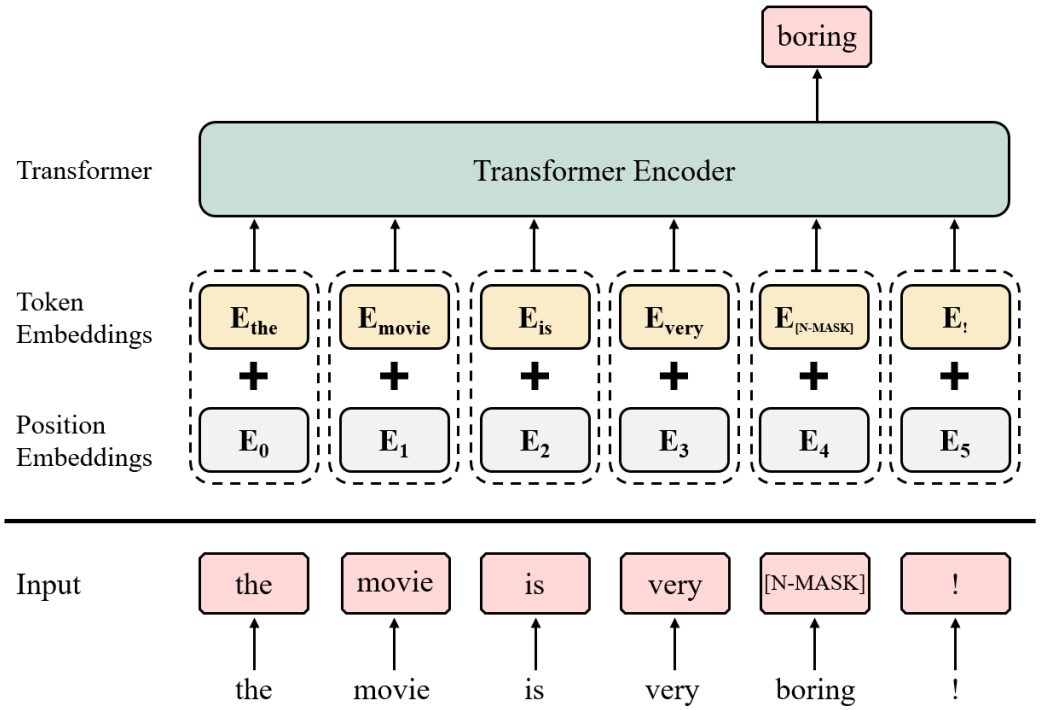

Figure 2. Model structure of the label-masked language model. [N-MASK] is a mask token containing negative label information.

3.2. Self-Supervised Contextual Data Augmentation

The proposed method learns context information through self-supervision using the given dataset. First, the dataset is duplicated N times for augmentation. Then, arbitrary words corresponding to τ% of each sentence are masked. τ is a parameter that determines how much to mask. Masked sentences, including their mask tokens, are used as inputs to the LMLM, which is then trained to predict the original words of the mask token. The preceding training process is considered a type of self-supervised learning. After this process of self-supervised learning is completed, a new sentence with a similar context of the same label as the existing sentence is created by randomly replacing one of the prediction

probabilities among the top-K words for all the mask tokens, with the LMLM and the dataset used during the training. K represents a number that determines whether to replace from the top few words. After the learning process is complete, we can perform data augmentation using the learned LMLM. The overview of the proposed method and test process are shown below in Figure 3.

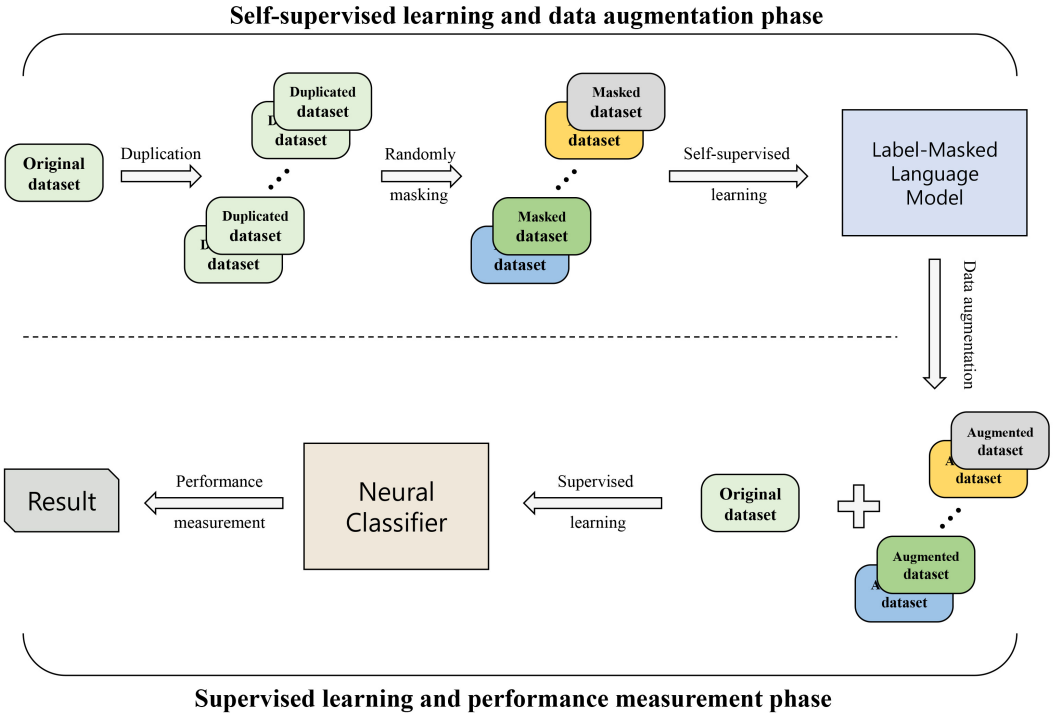

Figure 3. Overview of the proposed method. Self-supervised learning and data augmentation phase: We duplicate the original dataset, and mask random words for each dataset. After, the LMLM trains with a self-supervised learning method using various masked datasets, and then we perform data augmentation. Supervised learning and performance measurement phase: The neural classifier uses the original dataset and the augmented dataset to train for the text classification tasks.

The method can only replace words that are masked at the initial stage. Conversely, if we set N (the number of replicas made from the original dataset) as 1 and re-mask the original training data for each iteration on the self-supervised learning process, we can train a general MLM for a task-specific dataset. However, this technique takes a long time to converge.

4. Experiment and Results

4.1. Experiments Setup

We conducted an experiment on six text classification tasks including SST5, SST2, Subj, MPQA, MR, and TREC using LSTM-RNN and CNN-based neural classifiers.

4.2. Datasets

Six benchmark datasets were used in the experiments. In the training process, we used a development set to validate the performance of the model for various hyperparameters and to use the early stopping method. If the dataset had a development set, it was removed, and 10% of the training set, which was arbitrarily picked, was used as a new development set. In addition, 10-fold cross-validation was performed on data where test and training data were not split a priori. Below is a brief description of the benchmark dataset, and the summary statistics are in Table 1 [30].

- SST5: The Stanford Sentiment Treebank-5 is data for emotion classification tasks, which contains movie reviews with five labels (very positive, positive, neutral, negative, and very negative) [46].
- SST2: The Stanford Sentiment Treebank-2 is the same as SST5 but eliminates neutral reviews and has binary labels (positive and negative) [46].
- Subj: The subjectivity dataset is data for sentence classification tasks used to classify sentences with annotations based on whether they are subjective or objective [47].
- MPQA: The Multi-Perspective Question Answering dataset is an opinion polarity detection dataset consisting of short phrases. It has binary labels with positive and negative. [48].
- MR: Another Movie Reviews sentiment classification task dataset with binary labels (positive and negative) [49].
- TREC: The Text Retrieval Conference dataset consists of six question types and classifies the questions into different categories (abbreviation, entity, description, human, location, and numerical value) [50].

Table 1. Summary statistics for the datasets after tokenization. Class is the number of the target classes and length is average sentence length. The cross-validation indicates that the dataset was not divided between the training set and test set and thus 10-fold cross-validation was performed. The Stanford Sentiment Treebank-5 (SST5), the Stanford Sentiment Treebank-2 (SST2), the subjectivity (Subj), the Multi-Perspective Question Answering (MPQA), the Movie Reviews (MR),and the Text Retrieval Conference (TREC) datasets.

Dataset	Class	Length	Dataset Size	Vocabulary Size	Testset Size
SST5	5	18	11,855	17,836	2210
SST2	2	19	9613	16,185	1821
Subj	2	23	10,000	21,323	cross-validation
MPQA	2	3	10,606	6246	cross-validation
MR	2	20	10,662	18,765	cross-validation
TREC	6	10	5952	9592	500

4.3. Neural Classifier Architecture

We implemented LSTM-RNN and CNN-based neural classifiers in order to compare the performances of the contextual augmentation method [29] and our method.

The RNN based classifier consists of a single-layer LSTM, followed by a fully connected layer and the softmax function.

The CNN-based classifier uses convolutional filters of size ranges from three to five. The output from each filter was concatenated and then applied to the max-pooling over time, then fed to the fully connected layer with the ReLU as the activation function, followed by the softmax function. Both structures use dropout [17] and the Adam optimizer [51] to optimize their weights. We also terminated the training with an early stopping method to prevent the models from overfitting.

4.4. Hyperparameters

The hyperparameters of LMLM is shown in Table 2, and the training epoch was adjusted for each dataset. The learning rate, embedding dimension, number of the filters, and dropout rate of the classifier were selected using a grid search for each dataset with reference to the baseline hyperparameters [29]. The search space of the hyperparameters for the neural classifiers is shown in Table 3.

Table 2. The hyperparameters of label-masked language model (LMLM).

Hyperparameters	Value
Learning rate	0.0001
Embedding dimension	512
Number of the head (h)	4
Hidden dimension (d_k)	128
Feed forward dimension (d_{ff})	2048
Dropout rate	0.1

Table 3. The hyperparameter search space of neural classifiers.

Hyperparameters	Value
Learning rate	[0.001, 0.0003, 0.0001]
Embedding dimension	[300]
Hidden (filters) unit	[256, 512]
Dropout rate	[0.1, 0.2, 0.3, 0.4, 0.5]

In the data augmentation phase, we experimented with two combinations of N, indicating the number of replicas made from the original dataset, and K, the number determining whether to replace the top few words. The first combination was called "*Small*", (N = 2, K = 2), and does not replace the original word. The second was called "*Big*", (N = 5, K = 3), but the mask token can be replaced with the original word. In addition, the number of words that can be masked in each sentence ranges from 1 to 10, and the masking rate τ was tested at (0.05, 0.1, 0.2, 0.3, 0.4, 0.5). The reported accuracy was averaged over 20 models trained using different seeds and the best performance among the N, K, and τ combinations were recorded.

4.5. Baselines

We compare the following models to evaluate the performance of the proposed method.

- **RNN / CNN** Without data augmentation method.
- **w/synonym** A method of substituting random words with synonyms using WordNet [52].
- **w/context** A method of contextual augmentation for each word using bidirectional LM proposed by Kobayashi [29].
- **w/context+label** A method that adds a label condition to w/context [29].

4.6. Experiment Results

The quality of the generated data can be determined by the rate at which the words of the sentence are masked. If too few words are masked, they may not be different from the original sentences. If a large number of words are masked, words that can identify the context may be removed and the meaning of the original sentence may be lost. For this reason, the study used a rate between 0.05 and 0.5, but the masking rate of each dataset for data augmentation is a hyperparameter that the user must identify. We experimented with six datasets using the masking rates mentioned above. As a result, it was confirmed that the masking rates for obtaining the highest performance for each dataset differ from each other. That is, the distribution of words and the context representation may be different depending on the dataset. Thus, a task-specific masking rate should be determined. In addition, the number of words used in the sentence, the amount of data, and the average sentence length are all different. The dataset itself may affect how our method works. The masking rate is especially influenced by the length of sentences and the number of words used in each dataset. For this reason, the effect of the masking rate on each dataset is different. The performance of *Big* models according to the masking rates used for each dataset are shown below in Figures 4–9.

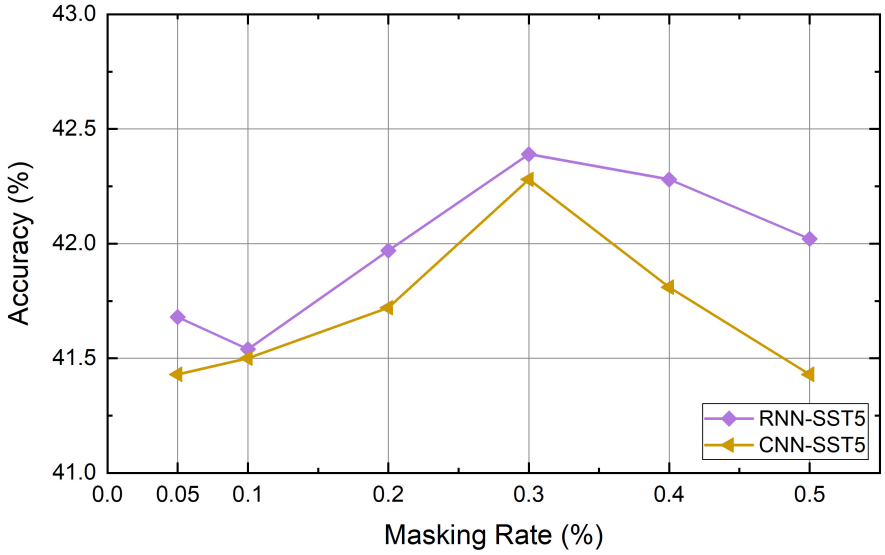

Figure 4. The effect of masking rate on the SST5 dataset. Recurrent neural network (RNN), convolutional neural network (CNN).

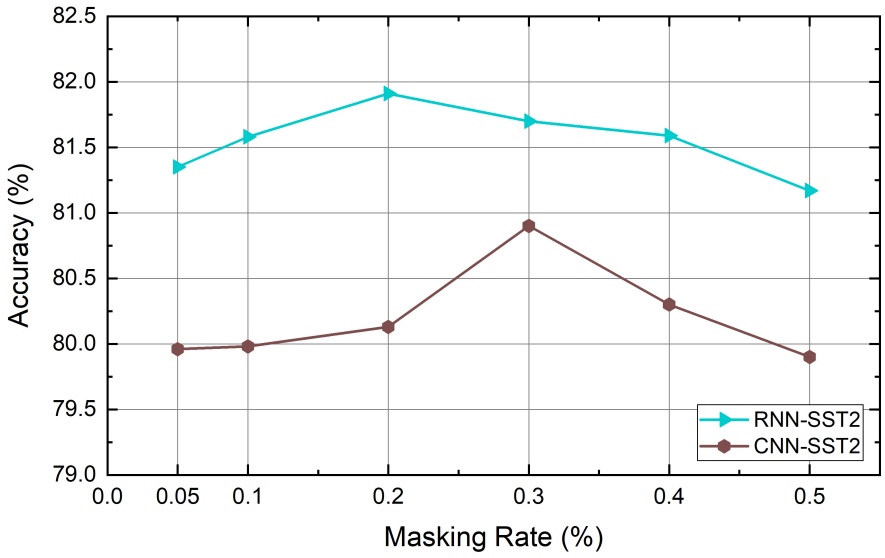

Figure 5. The effect of masking rate on the SST2 dataset.

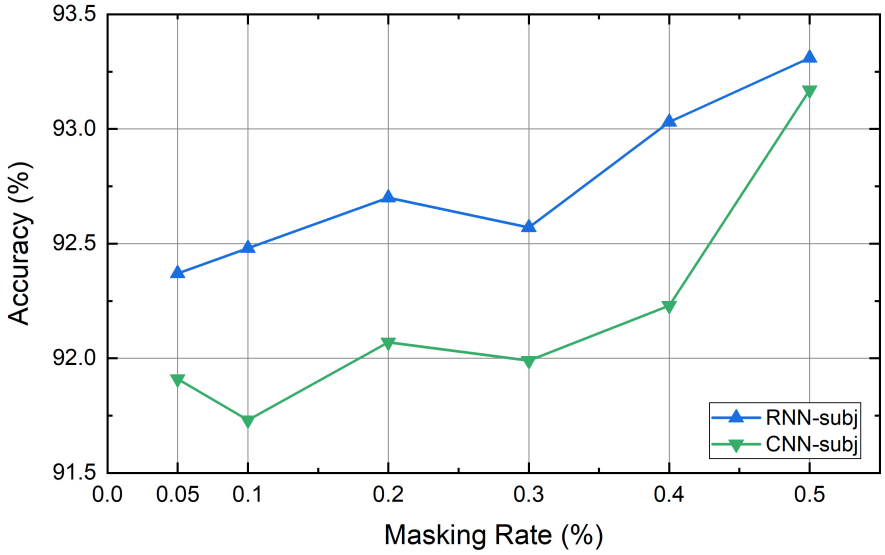

Figure 6. The effect of masking rate on the Subj dataset.

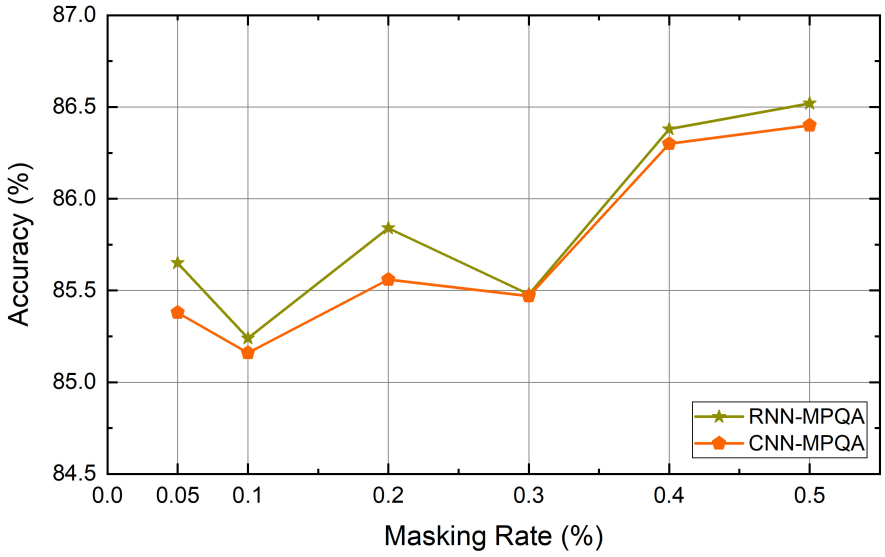

Figure 7. The effect of masking rate on the MPQA dataset.

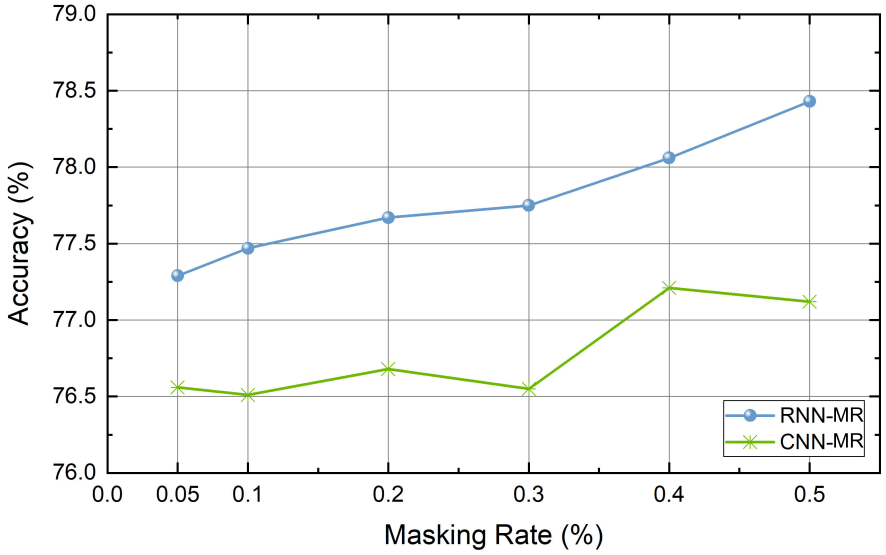

Figure 8. The effect of masking rate on the MR dataset.

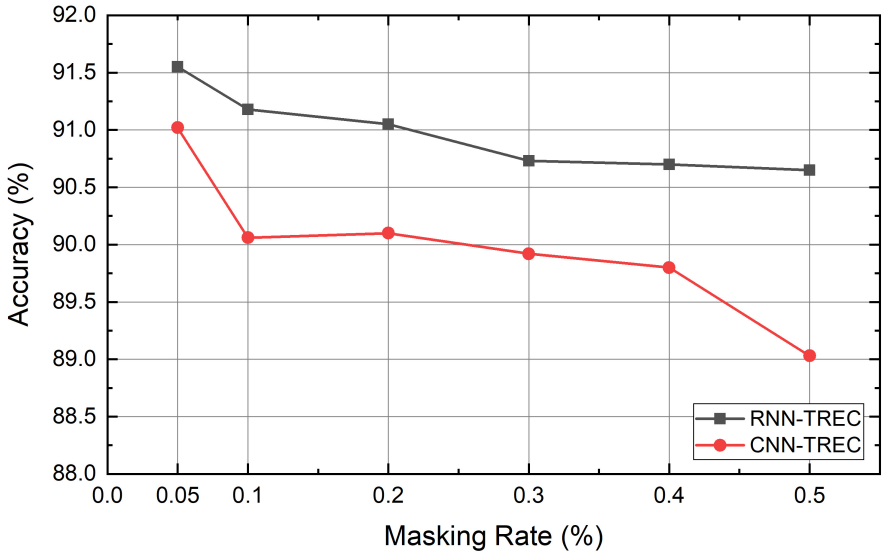

Figure 9. The effect of masking rate on the TREC dataset.

Table 4 lists the accuracy of the baseline and the proposed method. In Table 4, the values in parentheses represent the mask rate during data augmentation. The results show that our self-supervised contextual data augmentation method increases the performance of the model over most of the existing methods for various datasets. The performance enhancement is particularly noticeable in the RNN-based classifier when compared to the CNN-based classifier. Also, it can be seen that the performance has much improved in the case of the combination *Big*. We deem that *Big* can generate more various data than *Small* can.

In the case of the CNN-based classifier, it is better than the methods without data augmentation, but there are cases where it is similar to the method proposed by Kobayashi [29] or its performance is worse. However, our method is more efficient because Kobayashi's method requires a large amount of data and time for pre-training.

Kobayashi's method took about 20 h to pre-train shallow bidirectional LM using the Wikitext-103 dataset, and to fine-tune. In contrast, our method of self-supervised learning took less than two hours for each dataset. In this experiment, we used a Geforce RTX 2080 for training and used the deep learning frameworks Chainer [53] and Tensorflow [54] respectively. Although there is a difference in the framework, it shows that the proposed method can save a lot of time compared to the existing method.

Table 4. Accuracies and mask rates of the models on the different benchmark datasets. The values in parentheses represent the mask rate during data augmentation. The bold is the best performance.

Models	SST5	SST2	Subj	MPQA	MR	TREC	Avg.
RNN	40.2	80.3	92.4	86.0	76.7	89.0	77.43
w/ synonym	40.5	80.2	92.8	86.4	76.6	87.9	77.40
w/ context	40.9	79.3	92.8	86.4	77.0	89.3	77.62
+ label	41.1	80.1	92.8	86.4	77.4	89.2	77.83
Ours (Small)	42.0 (0.2)	81.5 (0.2)	93.0 (0.5)	86.1 (0.5)	78.3 (0.5)	91.1 (0.1)	78.67
Ours (Big)	**42.4** (0.3)	**81.9** (0.2)	**93.3** (0.5)	**86.5** (0.5)	**78.4** (0.5)	**91.6** (0.05)	**79.02**
CNN	41.3	79.5	92.4	86.1	75.9	90.0	77.53
w/ synonym	40.7	80.0	92.4	86.3	76.0	89.6	77.50
w/ context	41.9	**80.9**	92.7	**86.7**	75.9	90.0	78.02
+ label	42.1	80.8	93.0	**86.7**	76.1	90.5	78.20
Ours (Small)	41.7 (0.2)	79.9 (0.1)	92.5 (0.5)	86.2 (0.5)	76.9 (0.5)	90.9 (0.2)	78.02
Ours (Big)	**42.3** (0.3)	**80.9** (0.3)	**93.2** (0.5)	86.4 (0.5)	**77.2** (0.4)	**91.0** (0.05)	**78.50**

5. Discussion and Conclusions

In this paper, we propose a novel data augmentation method that takes context into consideration by using a masked language model for self-supervised learning. The experiment shows that our proposed method outperforms the conventional methods for most data. Our method, unlike the previous studies, simplifies the overall procedure by skipping pre-training and adopts LMLM to improve a bidirectional representation. In addition, our method is easy to use with any sentence that is labeled in various domains and tasks, without needing any linguistic knowledge or specific domain terminology.

This study has the limitation that it can only be used for labeled text data, and each dataset must find the masking rate that is appropriate for each. In addition, it takes longer to train the LMLM to create a generic LMLM for data augmentation on the task-specific dataset. In future works, we will focus on extending the current method for general and fast data augmentation that can be used for all kinds of NLP tasks beyond the problems of sentences with labels. We will also study robust methods regardless of the masking rate.

Author Contributions: Conceptualization, D.P.; methodology, D.P.; software, D.P.; investigation, D.P.; writing–original draft preparation, D.P.; writing–review and editing, D.P. and C.W.A.; funding acquisition, C.W.A.; supervision, C.W.A.

Funding: This work was supported by GIST Research Institute (GRI) grant funded by the GIST in 2019, and the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (NRF-2019R1I1A2A01057603).

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

MLM	Masked language model
CV	Computer vision
NLP	Natural language processing
LM	Language model
RNN	Recurrent neural network
CNN	Convolutional neural network
LSTM	Long short-term memory
ReLU	Rectified linear unit

References

1. Tan, M.; Le, Q. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In Proceedings of the International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; pp. 6105–6114.
2. Child, R.; Gray, S.; Radford, A.; Sutskever, I. Generating Long Sequences with Sparse Transformers. 2019. Available online: https://openai.com/blog/sparse-transformers (accessed on 10 November 2019).
3. Chen, L.C.; Zhu, Y.; Papandreou, G.; Schroff, F.; Adam, H. Encoder-decoder with atrous separable convolution for semantic image segmentation. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 801–818.
4. He, K.; Girshick, R.; Dollár, P. Rethinking imagenet pre-training. *arXiv* **2018**, arXiv:1811.08883.
5. Edunov, S.; Ott, M.; Auli, M.; Grangier, D. Understanding Back-Translation at Scale. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; pp. 489–500.
6. Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I. *Language Models Are Unsupervised Multitask Learners*. 2019. Available online: https://openai.com/blog/better-language-models/ (accessed on 10 November 2019).
7. Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J.; Salakhutdinov, R.; Le, Q.V. XLNet: Generalized Autoregressive Pretraining for Language Understanding. *arXiv* **2019**, arXiv:1906.08237.
8. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Florence, Italy, 28 July–2 August 2019; pp. 4171–4186.
9. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*; The MIT Press: Lake Tahoe, NV, USA, 2012; pp. 1097–1105.
10. Mikolov, T.; Karafiát, M.; Burget, L.; Černockỳ, J.; Khudanpur, S. Recurrent neural network based language model. In Proceedings of the Eleventh Annual Conference of the International Speech Communication Association, Makuhari, Chiba, Japan, 26–30 Setember 2010.
11. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems*; The MIT Press: Long Beach, CA, USA, 2017; pp. 5998–6008.
12. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z. Rethinking the inception architecture for computer vision. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 2818–2826.
13. Loshchilov, I.; Hutter, F. Sgdr: Stochastic gradient descent with warm restarts. *arXiv* **2016**, arXiv:1608.03983.
14. Pan, S.J.; Yang, Q. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.* **2009**, *22*, 1345–1359. [CrossRef]
15. Zoph, B.; Le, Q.V. Neural architecture search with reinforcement learning. *arXiv* **2016**, arXiv:1611.01578.
16. Balaji, A.; Allen, A. Benchmarking Automatic Machine Learning Frameworks. *arXiv* **2018**, arXiv:1808.06492.
17. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.

18. Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv* **2015**, arXiv:1502.03167.
19. Zhong, Z.; Zheng, L.; Kang, G.; Li, S.; Yang, Y. Random erasing data augmentation. *arXiv* **2017**, arXiv:1708.04896.
20. Zhang, X.; Zhao, J.; LeCun, Y. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*; Palais des Congrès de Montréal: Montréal, QC, Canada, 2015; pp. 649–657.
21. Wang, W.Y.; Yang, D. That's so annoying!!!: A lexical and frame-semantic embedding based data augmentation approach to automatic categorization of annoying behaviors using# petpeeve tweets. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, 17–21 September 2015; pp. 2557–2563.
22. Kafle, K.; Yousefhussien, M.; Kanan, C. Data augmentation for visual question answering. In Proceedings of the 10th International Conference on Natural Language Generation, Santiago de Compostela, Spain, 4–7 September 2017; pp. 198–202.
23. Wu, X.; Lv, S.; Zang, L.; Han, J.; Hu, S. Conditional BERT Contextual Augmentation. In *International Conference on Computational Science*; Springer: Seoul, Korea, 2019; pp. 84–95.
24. Xie, Q.; Dai, Z.; Hovy, E.; Luong, M.T.; Le, Q.V. Unsupervised data augmentation. *arXiv* **2019**, arXiv:1904.12848.
25. DeVries, T.; Taylor, G.W. Improved regularization of convolutional neural networks with cutout. *arXiv* **2017**, arXiv:1708.04552.
26. Zhang, H.; Cisse, M.; Dauphin, Y.N.; Lopez-Paz, D. mixup: Beyond empirical risk minimization. *arXiv* **2017**, arXiv:1710.09412.
27. Yun, S.; Han, D.; Oh, S.J.; Chun, S.; Choe, J.; Yoo, Y. CutMix: Regularization Strategy to Train Strong Classifiers with Localizable Features. In Proceedings of the International Conference on Computer Vision (ICCV), Seoul, Korea, 27 October–2 November 2019.
28. Gao, F.; Zhu, J.; Wu, L.; Xia, Y.; Qin, T.; Cheng, X.; Zhou, W.; Liu, T.Y. Soft Contextual Data Augmentation for Neural Machine Translation. In Proceedings of the 57th Conference of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; pp. 5539–5544.
29. Kobayashi, S. Contextual Augmentation: Data Augmentation by Words with Paradigmatic Relations. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Melbourne, Australia, 19 July 2018; pp. 452–457.
30. Kim, Y. Convolutional Neural Networks for Sentence Classification. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014; pp. 1746–1751.
31. Aytar, Y.; Pfaff, T.; Budden, D.; Paine, T.; Wang, Z.; de Freitas, N. Playing hard exploration games by watching youtube. In *Advances in Neural Information Processing Systems*; Palais des Congrès de Montréal: Montréal, QC, Canda, 2018; pp. 2930–2941.
32. Pathak, D.; Agrawal, P.; Efros, A.A.; Darrell, T. Curiosity-driven exploration by self-supervised prediction. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, Honolulu, HI, USA, 21–26 July 2017; pp. 16–17.
33. Kahn, G.; Villaflor, A.; Ding, B.; Abbeel, P.; Levine, S. Self-supervised deep reinforcement learning with generalized computation graphs for robot navigation. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018; pp. 1–8.
34. Kolesnikov, A.; Zhai, X.; Beyer, L. Revisiting Self-Supervised Visual Representation Learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019; pp. 1920–1929.
35. Gidaris, S.; Singh, P.; Komodakis, N. Unsupervised Representation Learning by Predicting Image Rotations. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–2 May 2018.
36. Kim, D.; Cho, D.; Yoo, D.; Kweon, I.S. Learning Image Representations by Completing Damaged Jigsaw Puzzles. In Proceedings of the 2018 IEEE Winter Conference on Applications of Computer Vision (WACV), Lake Tahoe, NV, USA, 15 March 2018; pp. 793–802, doi:10.1109/WACV.2018.00092. [CrossRef]
37. Lample, G.; Conneau, A. Cross-lingual language model pretraining. *arXiv* **2019**, arXiv:1901.07291.

38. Pathak, D.; Krahenbuhl, P.; Donahue, J.; Darrell, T.; Efros, A.A. Context encoders: Feature learning by inpainting. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 2536–2544.
39. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
40. Fadaee, M.; Bisazza, A.; Monz, C. Data Augmentation for Low-Resource Neural Machine Translation. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, Vancouver, BC, Canada, 30 July–4 August 2017; pp. 567–573.
41. Ko, T.; Peddinti, V.; Povey, D.; Khudanpur, S. Audio augmentation for speech recognition. In Proceedings of the Sixteenth Annual Conference of the International Speech Communication Association, Dresden, Germany, 6–10 September 2015.
42. Hannun, A.; Case, C.; Casper, J.; Catanzaro, B.; Diamos, G.; Elsen, E.; Prenger, R.; Satheesh, S.; Sengupta, S.; Coates, A.; et al. Deep speech: Scaling up end-to-end speech recognition. *arXiv* **2014**, arXiv:1412.5567.
43. Mallidi, S.H.; Hermansky, H. Novel neural network based fusion for multistream ASR. In Proceedings of the 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Shanghai, China, 20–25 March 2016; pp. 5680–5684.
44. Park, D.S.; Chan, W.; Zhang, Y.; Chiu, C.C.; Zoph, B.; Cubuk, E.D.; Le, Q.V. Specaugment: A simple data augmentation method for automatic speech recognition. *arXiv* **2019**, arXiv:1904.08779.
45. Taylor, W.L. "Cloze procedure": A new tool for measuring readability. *J. Bull.* **1953**, *30*, 415–433. [CrossRef]
46. Socher, R.; Perelygin, A.; Wu, J.; Chuang, J.; Manning, C.D.; Ng, A.; Potts, C. Recursive deep models for semantic compositionality over a sentiment treebank. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, Seattle, WA, USA, 18–21 October 2013; pp. 1631–1642.
47. Pang, B.; Lee, L. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics, Barcelona, Spain, 21–26 July 2004; p. 271.
48. Wiebe, J.; Wilson, T.; Cardie, C. Annotating expressions of opinions and emotions in language. *Lang. Resour. Eval.* **2005**, *39*, 165–210. [CrossRef]
49. Pang, B.; Lee, L. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, University of Michigan, Ann Arbor, MI, USA, 25–30 June 2005; pp. 115–124.
50. Li, X.; Roth, D. Learning question classifiers. In *Proceedings of the 19th International Conference on Computational linguistics—Volume 1*; Association for Computational Linguistics: Philadelphia, PA, USA, 2002; pp. 1–7.
51. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
52. Miller, G.A. WordNet: A lexical database for English. *Commun. ACM* **1995**, *38*, 39–41. [CrossRef]
53. Tokui, S.; Oono, K.; Hido, S.; Clayton, J. Chainer: A next-generation open source framework for deep learning. In Proceedings of the Workshop on Machine Learning Systems (LearningSys) in the Twenty-Ninth Annual Conference on Neural Information Processing Systems (NIPS), Montréal, QC, Canada, 8–13 December 2015; Volume 5, pp. 1–6.
54. Abadi, M.; Barham, P.; Chen, J.; Chen, Z.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Irving, G.; Isard, M.; et al. TensorFlow: A system for large-scale machine learning. In Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16), Savannah, GA, USA, 2–4 November 2016; pp. 265–283.