applied sciences

Article

FrameAugment: A Simple Data Augmentation Method for Encoder–Decoder Speech Recognition

Seong-Su Lim [1] and Oh-Wook Kwon [2,*]

[1] Major in Control and Robot Engineering, Chungbuk National University, Cheongju 28644, Korea; ansun369@naver.com
[2] Department of Intelligent Systems and Robotics, Chungbuk National University, Cheongju 28644, Korea
[*] Correspondence: owkwon@cbnu.ac.kr

Abstract: As the architecture of deep learning-based speech recognizers has recently changed to the end-to-end style, increasing the effective amount of training data has become an important issue. To tackle this issue, various data augmentation techniques to create additional training data by transforming labeled data have been studied. We propose a method called FrameAugment to augment data by changing the speed of speech locally for selected sections, which is different from the conventional speed perturbation technique that changes the speed of speech uniformly for the entire utterance. To change the speed of the selected sections of speech, the number of frames for the randomly selected sections is adjusted through linear interpolation in the spectrogram domain. The proposed method is shown to achieve 6.8% better performance than the baseline in the WSJ database and 9.5% better than the baseline in the LibriSpeech database. It is also confirmed that the proposed method further improves speech recognition performance when it is combined with the previous data augmentation techniques.

Keywords: data augmentation; end-to-end speech recognition; frame rate

check for updates

Citation: Lim, S.-S.; Kwon, O.-W. FrameAugment: A Simple Data Augmentation Method for Encoder–Decoder Speech Recognition. *Appl. Sci.* **2022**, *12*, 7619. https://doi.org/10.3390/app12157619

Academic Editor: Douglas O'Shaughnessy

Received: 12 May 2022
Accepted: 18 July 2022
Published: 28 July 2022

1. Introduction

Deep learning [1] has recently shown remarkable performance in many fields including speech recognition. However, the performance of the deep learning network architecture is greatly affected by the amount of training data. Therefore, researchers have investigated increasing the effective size of training data by utilizing unlabeled data for learning [2–5], or perturbing labeled data and augmenting it to training data [6,7].

As up-to-date speech recognizers have begun to be changed from a hybrid network architecture to a deep learning network architecture [8], much effort has been made to increase the amount of training data effectively. Due to such effort, several data augmentation techniques have been proposed. For example, the speed perturbation [9] technique complements training data by using the data obtained to resample the labeled data at several different speeds. The SpecAugment [10] technique adds speech data obtained by randomly overlaying a spectrogram with binary masks. The SpecSwap [11] technique obtains new speech data by switching two parts of the spectrogram.

We propose a data augmentation method that randomly changes the speed of randomly chosen sections of an utterance. Compared with the speed perturbation technique that changes the speed of the entire utterance at a single fixed speed, the proposed method changes the speed of an utterance at a different speed for every section, and thus, can give more speed variability to the augmented speech data. The speed perturbation technique operates in the waveform domain, whereas the proposed method works in the spectrogram domain. The sections where the speed is changed are randomly selected in a similar manner to the SpecAugment technique, and the speed of the selected sections is changed through linear interpolation. We validate the effectiveness of the proposed method reliably using the WSJ database [12] and LibriSpeech [13] 100 h database.

In Section 2, we describe the network architecture used as a baseline speech recognizer, and review the previous data augmentation methods. In Section 3, we describe the proposed method. In Section 4, we show the experimental results using the WSJ and LibriSpeech databases, optimize hyperparameters, and provide the analysis results and discussion. In Section 5, conclusions are drawn.

2. Related Works

2.1. Network Architecture

As speech recognizers change from a hybrid network architecture to a deep learning network architecture, many deep learning network architectures have been proposed for speech recognition [14,15]. Speech recognizers generally adopt an encoder–decoder network architecture. In the encoder–decoder architecture, the encoder receives the entire utterance and outputs the features of the utterance, and the decoder outputs the result by using the feature vector input from the encoder and the previous result. As shown in Figure 1, the Transformer [16] model is used as the baseline encoder–decoder network architecture except that the input embedding layer is replaced by a subsampling layer to compress the input spectrogram. The encoder of the Transformer learns the relationship among spectrogram data in different positions through self-attention and outputs feature vectors. Because self-attention does not perform regression as LSTM (long short-term memory) [17] does, parallel processing is possible. The decoder of the Transformer uses self-attention like an encoder. However, whereas the encoder's self-attention learns the positional relationship of the spectrogram, the decoder's self-attention learns the relationship among the output units. Because the rear part of the currently predicted unit cannot be known, the decoder uses masked-self-attention in the learning process, which learns by masking the rear part of the prediction unit. In the subsequent multi-head attention sub-layer, the decoder learns the relationship between input and output.

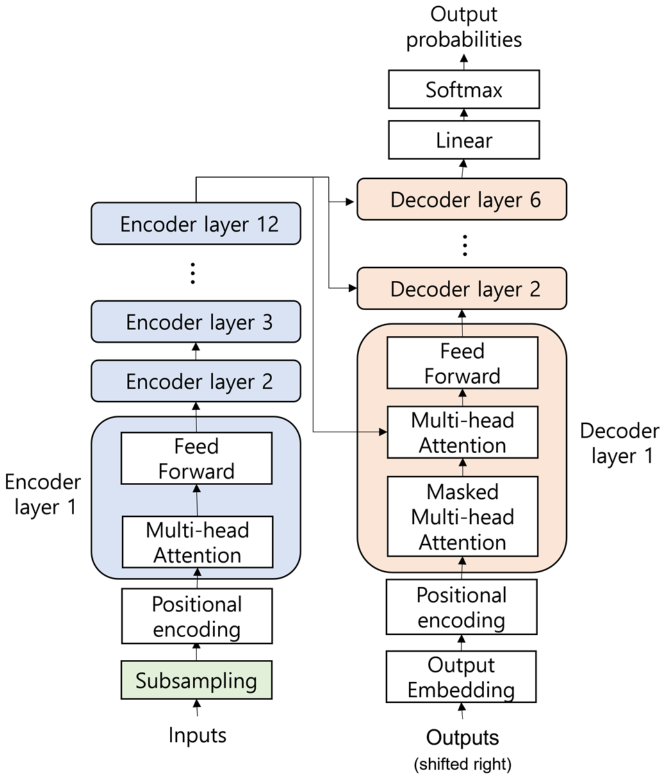


Figure 1. Baseline encoder-decoder network architecture.

In the encoder–decoder network architecture, it was difficult to align at the beginning of the learning process. To solve this problem, CTC (connectionist temporal classification) [15] was combined with the encoder from the MTL (multi-task learning) [18] method. The combined architecture was shown to improve learning speed and speech recognition performance compared to the single model by alleviating the alignment problem [19].

The total loss of the combined architecture is calculated as follows:

$$L_{MTL} = \tau L_{CTC} + (1 - \tau) L_{Attention} \tag{1}$$

where τ is a constant greater than or equal to 0 and less than or equal to 1, L_{CTC} is CTC loss and $L_{Attention}$ is attention loss obtained from the Transformer, respectively.

2.2. Speed Perturbation

Speed perturbation [9] is a method of data augmentation that works by changing the speed of the training data. For speed perturbation, we used Sox [20] to resample the speech part of the training data at 90% and 110% of the original rate. The resampled training data obtained from the existing training data were added to the original training data, and the size of the final training data became 3 times the capacity of the original training data. That is, we created 3 times the training data in speed perturbation experiments. Hereinafter, in this paper, this is called 3-fold speed perturbation.

2.3. SpecAugment

SpecAugment [10] modifies a spectrogram through three kinds of deformations: time warping, frequency masking, and time masking. Figure 2 shows a spectrogram to which SpecAugment is applied. We assume that the spectrogram has the length of the time axis L and the length of the frequency axis V. Time warping is a deformation that occurs by selecting a random point in the center of the frequency axis of the spectrogram in time $(W, L - W)$ and warping it left or right by w chosen from the uniform distribution, from 0 to the time warp parameter W. Frequency masking is a deformation of masking frequency channels $[f_0, f_0 + f)$ and by selecting a frequency channel number f within the frequency $[0, V - f]$. The f is chosen from a uniform distribution from 0 to the frequency parameter F. Time masking is a deformation of masking the frame number $[t_0, t_0 + t)$ by selecting a frame number within time $[0, L - t]$. The time t is chosen from a uniform distribution, from 0 to the time parameter T.

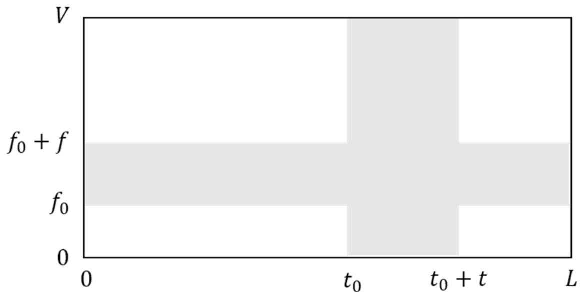


Figure 2. SpecAugment applied to spectrogram. The sections to which frequency masking was applied $[f_0, f_0 + f)$ and time masking was applied $[t_0, t_0 + t)$ are shown in gray.

2.4. Interpolation

We adopted the simplest interpolation method, linear interpolation [21], to change the frame rate in the spectrogram. Linear interpolation has the advantage that it is simple and fast to calculate. The linear interpolation method finds the interpolated value by linearly connecting two given points. Given two points (t_0, f_0), (t_1, f_1), the linear interpolation method calculates the interpolated value as follows:

$$f = f_0 + (f_1 - f_0)\frac{t - t_0}{t_1 - t_0} \tag{2}$$

3. Proposed Method

The proposed method in this paper aims to increase the effective amount of learning data by changing the labeled data in the same way as the existing data augmentation techniques. However, unlike speed perturbation that changes the speed of the entire speech, the proposed method has the advantage of obtaining more learning data because data is augmented by changing the speed of a few sections of speech.

As shown in Figure 3, the proposed method consists of four steps and is described sequentially in the next section.

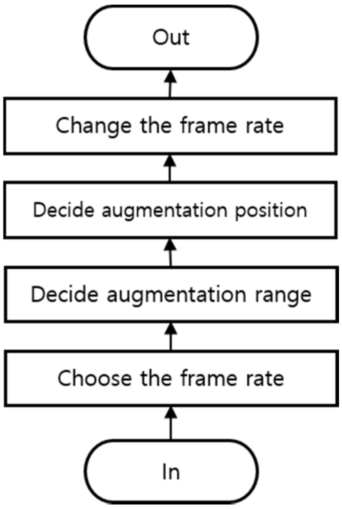


Figure 3. Flow chart of the proposed method.

3.1. Frame Rate

Since the proposed method transforms the spectrogram, the speed of speech is changed by changing the frame rate through linear interpolation. For example, the frame rate can be halved by making the existing 6 frames into 3 frames, and can also be reduced to 2/3 by making 4 frames. In informal listening tests, we found that speech is intelligible when the frame rate is doubled or halved. Therefore, all subsequent experiments were performed by changing the speed of speech between 1/2 and 2. In addition, the average frame rate is set to 1 so that the average length of the augmented data is the same as the average length of the input data. The frame rate s is determined as follows, and then it is rounded to the first decimal place for simplicity:

$$s \sim \text{Uniform}[S_1, S_2], \frac{S_1 + S_2}{2} = 1 \quad (3)$$

where $\text{Uniform}[a, b]$ is the uniform distribution from a to b, and S_1 and S_2 denote the lower and upper bounds of the frame rate, respectively.

3.2. Augmentation Range

Since it does not change the speed of the entire utterance, it is necessary to determine the range in which the speed is to be changed. When deciding the augmentation range of the utterance, the range was randomly determined, as in SpecAugment:

$$n \sim \text{Uniform } [0, N] \quad (4)$$

where N is a parameter representing the maximum range which is determined later through experiments.

3.3. Augmentation Position

Since we also change the speed of a section of the utterance, it is necessary to determine the augmentation range as well as the position to be augmented. The augmentation position was determined at random, the same as for the augmentation range. The augmentation position p is determined as:

$$p \sim \text{Uniform}[0, L - n] \tag{5}$$

Since the augmentation position should not exceed the end of the utterance, the start position of the augmentation range is set by subtracting the augmentation range n from the end of the utterance. Figure 4 is a spectrogram to which the proposed method is applied according to the augmentation range n.

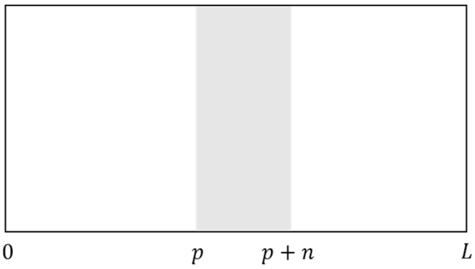


Figure 4. Proposed method applied to base input spectrogram. The section to which the proposed method was applied is shown in gray.

We obtained variable positional sections through the steps outlined in Sections 3.2 and 3.3. These methods are similar to those used in the time masking of SpecAugment. However, unlike SpecAugment, the speed of the selected augmentation range is changed, as is explained in Section 3.4.

3.4. Linear Interpolation

The proposed method changes the speed of speech by changing the frame rate. When changing the frame rate, the existing frame value is used to calculate an alternate frame value. For example, when the frame rate is changed to 0.6, in the augmentation sections, the existing 5 frames are replaced with 3 frames calculated through linear interpolation. The following shows the frame rate change process in the augmentation sections of the proposed method.

The number of frames to be extracted, a is obtained for each augmentation section as follows:

$$a = s \times n \tag{6}$$

$$t_k = t_0 + \frac{k}{s},\ k = 0, 1, \cdots, a - 1 \tag{7}$$

where t_0 is the position of the existing frame, and t_k is the position of the replacement frame. After obtaining the positions t_k of the replacement frame value, the replacement frame value is obtained by putting it into the feature v that is correlated with t_k. Finally, the existing feature frames of the feature vector $v[t]$ is replaced by the replacement frames of the feature vector $v[t_k]$. Additionally, each frame feature does not affect each other, i.e., if the number of the feature vector v is L, then features $\{v_1, v_2, \cdots, v_L\}$ are independent of each other. Figure 5 shows how replacement frames are calculated through linear interpolation.

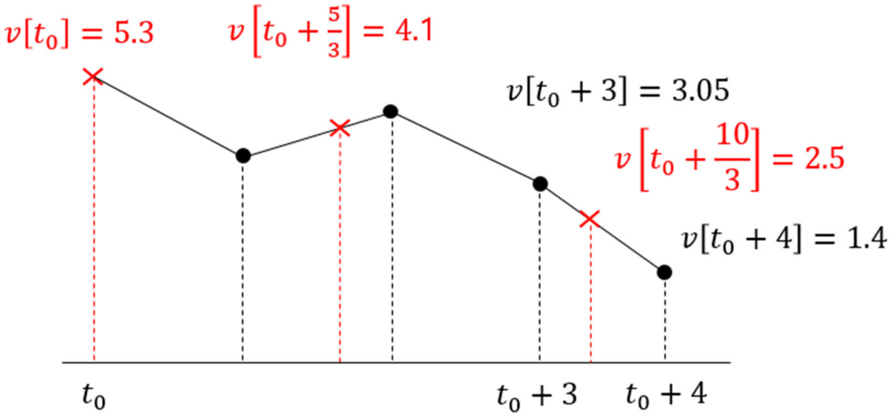

$v[t_0] = 5.3$
$v\left[t_0 + \frac{5}{3}\right] = 4.1$
$v[t_0 + 3] = 3.05$
$v\left[t_0 + \frac{10}{3}\right] = 2.5$
$v[t_0 + 4] = 1.4$
t_0
$t_0 + 3$
$t_0 + 4$

Figure 5. Method of calculating replacement frames. The red cross indicates replacement frame values and the black bullet indicates original frame values when $s = \frac{6}{10}$, respectively.

4. Results

In the proposed method, the degree of augmentation is determined by several parameters. Therefore, an experiment was conducted to find the best parameters.

For the reliability of our experiments, two English databases were used: WSJ and LibriSpeech. The WSJ database consists of 81 h of training data, and consists of validation data "dev93" and evaluation data "eval92". The LibriSpeech database consists of 960 h of training data, and consists of validation data "dev-clean" and evaluation data "test-clean". In this paper, when learning to use LibriSpeech, only 100 h of learning data was used to reduce the computing resource requirement. In addition, we used the character-based recognition unit for the WSJ database, and used the SentencePiece recognition unit [22] obtained by using the unigram language model [23] for the LibriSpeech database.

In this paper, experiments were conducted using the ESPnet toolkit [24] and the combined Transformer-CTC structure was used as the model. We used 12 encoder layers and 6 decoder layers, where each multi-head attention sub-layer was configured to have 4 heads. Additionally, we set dropout [25] to 0.1, and smoothing [26] to 0.1. For training, we set the maximum sequences in a minibatch ("batch-size") [24] to 16 with the number of gradient accumulation ("accum-grad") 4 in WSJ, and set the maximum bins in a minibatch ("batch-bins") [24] to 2,996,000 with "accum-grad" 16 in LibriSpeech. After 100 epochs of learning, in the WSJ database, we used the model that averaged the top 10 models with the highest accuracy of validation data. In the LibriSpeech database, we used the model that averaged the top five models.

4.1. Augmentation Range

Since the proposed method changes the speed of some sections of an utterance, we used the degree of augmentation according to the range of the augmentation section. We experimented with changing the maximum number of frames to find the parameter showing the optimal performance. In addition, experiments were conducted when the proposed augmentation method was repeatedly applied while the sections did not overlap. If the augmentation range n was longer than the length of the utterance, the entire utterance was augmented. Table 1 shows the experimental results for the WSJ database.

When the maximum augmentation range N is 500 frames, it shows the highest performance on average in the validation data. Repetition of the augmentation method did not significantly affect the recognition performance. Figure 6 is the graphical plot corresponding to Table 1.

The average frame length of the WSJ database is about 900 frames. Performance was best at 500 frames; thus, the performance was best when it was about 60% of the utterance length on average. Table 2 shows the experimental results for the LibriSpeech database.

Table 1. WER (%) results of WSJ database according to the maximum augmentation range N when $s \sim \text{Uniform}[0.9, 1.1]$.

N	Number of Repetitions	Dev93	Eval92
0	0	7.88	5.02
100	1	7.81	4.78
	2	7.35	4.86
200	1	7.27	4.45
	2	7.51	4.91
300	1	7.66	5.07
	2	7.27	4.43
400	1	7.55	4.27
	2	7.20	5.05
500	1	7.03	4.75
	2	6.97	5.00
600	1	7.25	4.73
	2	7.31	5.12
700	1	7.49	4.50
	2	7.26	4.92

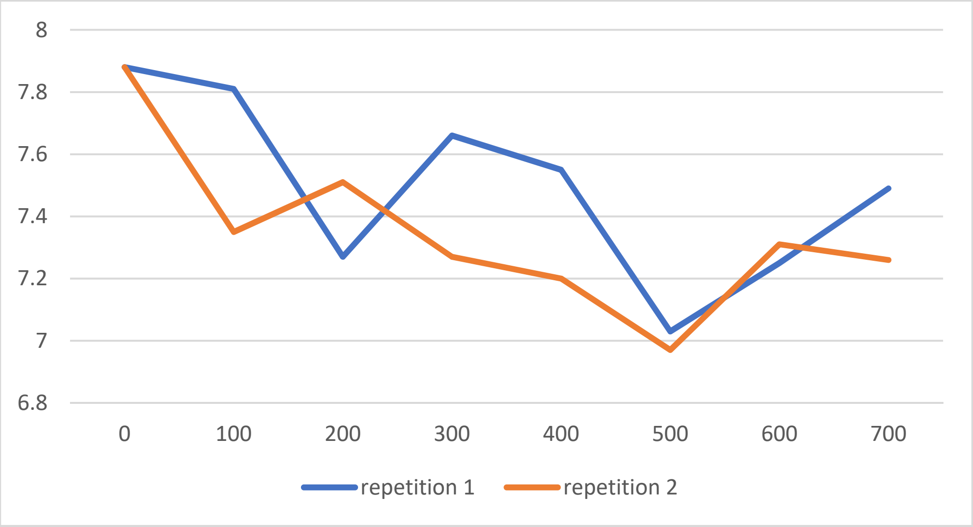


Figure 6. WER results of WSJ database according to augmented frame range. The horizontal axis denotes the maximum augmentation range and the vertical axis denotes WER.

Table 2. WER (%) results of LibriSpeech database according to the maximum augmentation range N when $s \sim \text{Uniform}[0.9, 1.1]$.

N	Dev-Clean	Test-Clean
0	6.62	7.33
200	6.71	7.44
400	6.74	7.37
600	6.28	7.01
800	6.46	7.12
1000	6.31	6.95
1200	6.26	6.96
1400	6.61	7.16

The average frame length of the LibriSpeech database is about 1500 frames. Performance was best at 1200 frame; thus, the performance was best when it was about 80% of the utterance length on average.

Through the results of the LibriSpeech and the WSJ database, the recognition result did not change according to the absolute number of frames, but according to the ratio of the augmentation section in the utterance.

4.2. Frame Rate

The proposed method changes the degree of augmentation by changing the speed in the augmentation section. As described above, the method of changing the speed uses linear interpolation to change the number of frames. In addition, we experimented with the maximum frame rate range of 0.5–1.5 in order to transform at an audible speed. Speed s is rounded to the first digit after the decimal point. Tables 3 and 4 show the performance comparison according to frame rate. Even if the frame rate is randomly selected from a set of the values consisting of the minimum range, 1.0, and the maximum range, there is no significant performance difference. For example, comparing $s = 0.5$–1.5 and $s = 0.5,\ 1.0,\ 1.5$, we observed no consistent WER changes.

Table 3. WER (%) results of WSJ database according to frame rate s when $n = 500$.

s	Dev93	Eval92
0	7.88	5.02
0.9–1.1	7.03	4.75
0.7–1.3	7.38	4.82
0.7, 1.0, 1.3	7.07	4.87
0.5–1.5	7.24	4.94
0.5, 1.0, 1.5	7.04	4.70

Table 4. WER (%) results of LibriSpeech database according to frame rate s when $n = 1200$.

s	Dev-Clean	Test-Clean
0.9–1.1	6.26	6.96
0.7–1.3	5.95	6.60
0.7, 1.0, 1.3	6.15	6.62
0.5–1.5	6.06	6.68
0.5, 1.0, 1.5	5.93	6.74

In the WSJ database, there was no significant difference in performance even by changing the range of the frame rate, but in the case of the LibriSpeech database, changing the range of the frame rate showed significant performance improvement. Different results were obtained for the two databases, and additional confirmation was performed through validation loss in Figure 7. In the WSJ database, there was no difference in the validation loss between 0.9–1.1 and 0.5–1.5 for the frame rate, but in the LibriSpeech database, the larger the range of the frame rate, the lower the validation loss value.

Optimal parameters of s and n were tested through experiments. The best parameters s were 0.5–1.5 and the best parameter N was 500 in the WSJ database and 1200 in the LibriSpeech database.

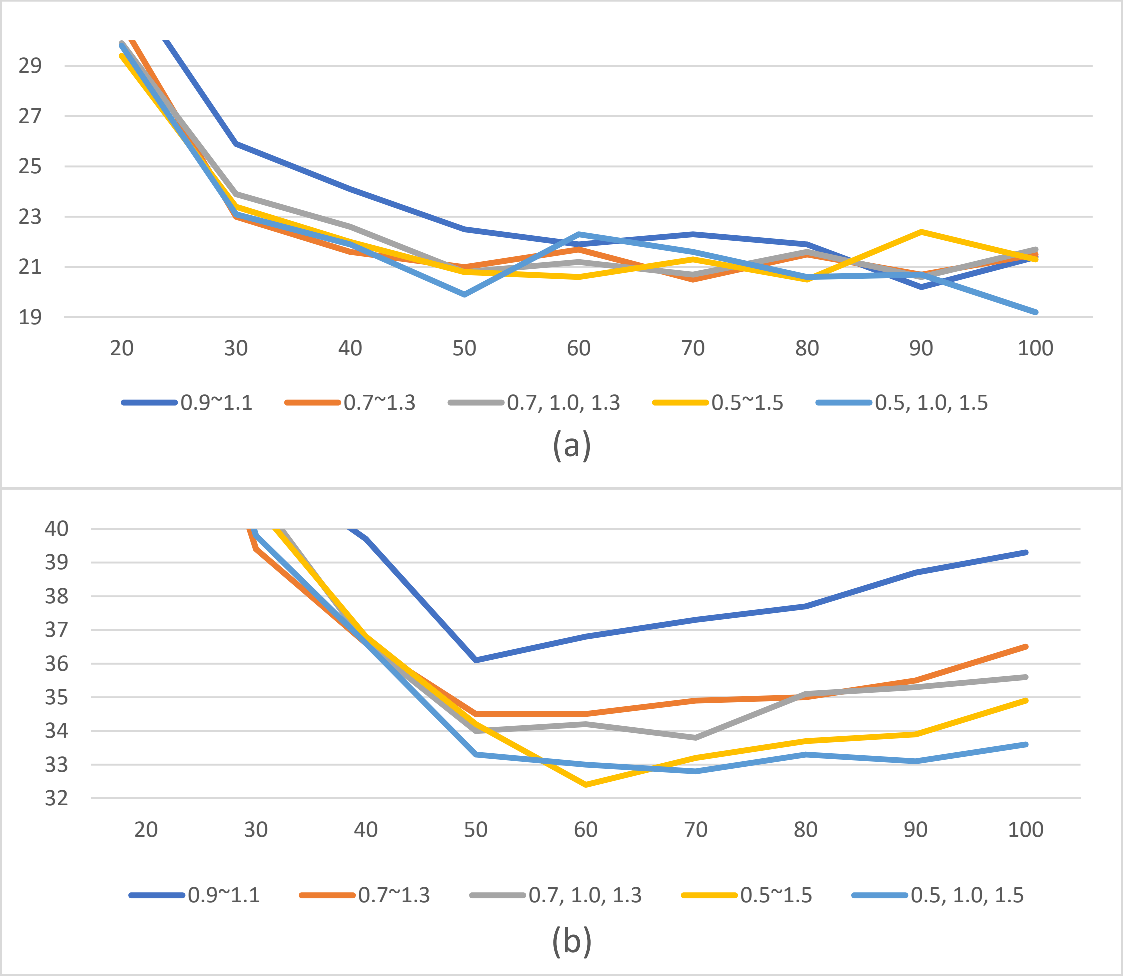


Figure 7. Validation loss of WSJ (**a**) and LibriSpeech (**b**) databases by epoch. The horizontal axis denotes the number of epochs and the vertical axis denotes validation loss.

4.3. Determining the Maximum Augmentation Range

Through the experiment, it was confirmed that the speech recognition performance of the proposed method depends on the ratio of the augmentation section in the entire utterance. The main reason for the difference in the optimal *N* for the WSJ and LibriSpeech databases is that the average utterance length of the databases is different.

Therefore, we redefined the value of *N* considering the length of each utterance. For example, if it is assumed that the length of the utterance is 100 and the predefined ratio is 0.8, then the maximum augmentation range becomes $N = 80$. That is, *N* changes according to the length of each utterance as follows:

$$N = L \times ratio \tag{8}$$

where *ratio* is a rational number less than 1.

Table 5 shows the experimental results in which the augmentation sections were determined to be proportional to the utterance length. In the flexible N case, the augmentation ratio was set to 70%, the average of 60% and 80%, which performed best in the WSJ and LibriSpeech databases.

The experimental results thus far imply that the augmentation range and speed in FrameAugment control the degree of augmentation and, consequently, improve recognition performance. Therefore, as in SpecAugment, the tuning of augmentation parameters is important. In order to cope with diverse kinds of databases as far as possible and not just a specific database, the augmentation range was set at random, the maximum augmentation range was also decided considering the utterance length, and the speed was also set at random within a given range.

Table 5. WER (%) results according to the maximum augmentation range. For the flexible N case, the ratio was 0.7 and the frame rate range was 0.5–1.5 in the proposed method.

Database	N	Dev93/Dev-Clean	Eval92/Test-Clean
WSJ	0	7.88	5.02
	Fixed $(N = 500)$	7.24	4.94
	Flexible $(N = L \times 0.7)$	6.89	4.68
LibriSpeech	0	6.62	7.33
	Fixed ($N = 1200)$	6.06	6.68
	Flexible $(N = L \times 0.7)$	6.14	6.63

4.4. Comparison with the Existing Data Augmentation Methods

SpecAugment and speed perturbation are widely used as a data augmentation method. We compared the performance of the previous data augmentation and the proposed method. For SpecAugment, we set the parameters to W = 5, F = 30, T = 40, and n_mask = 2. Tables 6 and 7 show the performance of the existing data augmentation methods and the proposed method in WSJ and LibriSpeech, respectively.

Table 6. WER (%) results of WSJ database according to data augmentation methods.

Data Augmentation	Dev93	Eval92
None	7.88	5.02
SpecAugment	7.23	4.87
Speed perturbation	7.24	4.93
Proposed method	6.89	4.68

Table 7. WER (%) results of LibriSpeech database according to data augmentation methods.

Data Augmentation	Dev-Clean	Test-Clean
None	6.62	7.33
SpecAugment	6.23	6.96
Speed perturbation	6.10	6.74
Proposed method	6.14	6.63

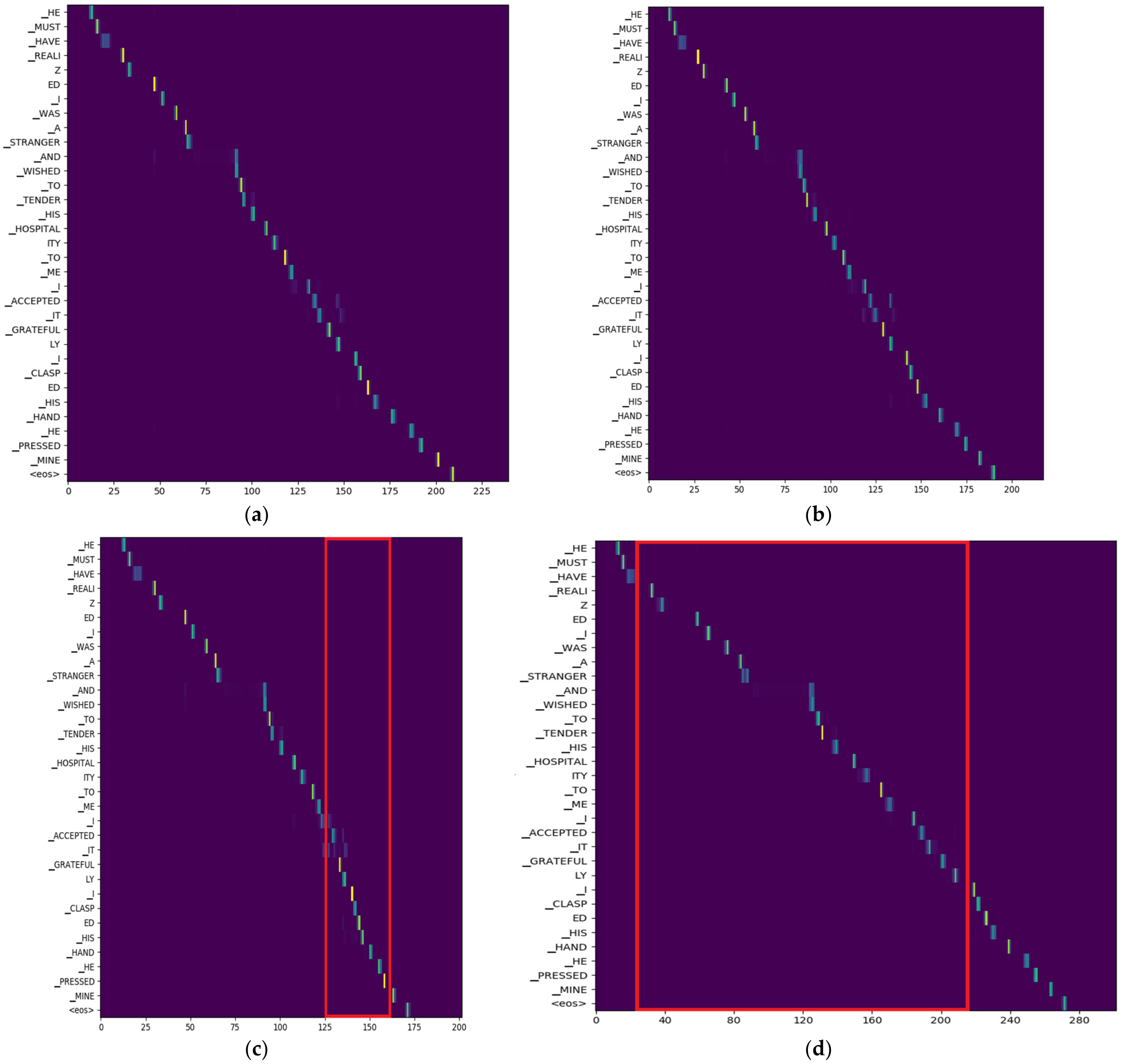
Table 8 shows the performance of incrementally combining the existing data augmentation methods and the proposed method in LibriSpeech. The combination of SpecAugment and speed perturbation with the proposed method of 6.01% improved performance by up to 13.6% over the baseline of 6.96%. In addition, the proposed method was shown to further reduce WER from 6.26 to 6.01% compared with the combination of the two previous data augmentation methods (SpecAugment and speed perturbation), which means a relative performance improvement of 4%.

Table 8. WER (%) results of LibriSpeech database when incrementally combined with the previous data augmentation methods.

Data Augmentation	Dev-Clean	Test-Clean
SpecAugment	6.23	6.96
+Speed perturbation	5.83	6.26
+Proposed method	5.53	6.01

4.5. Discussion

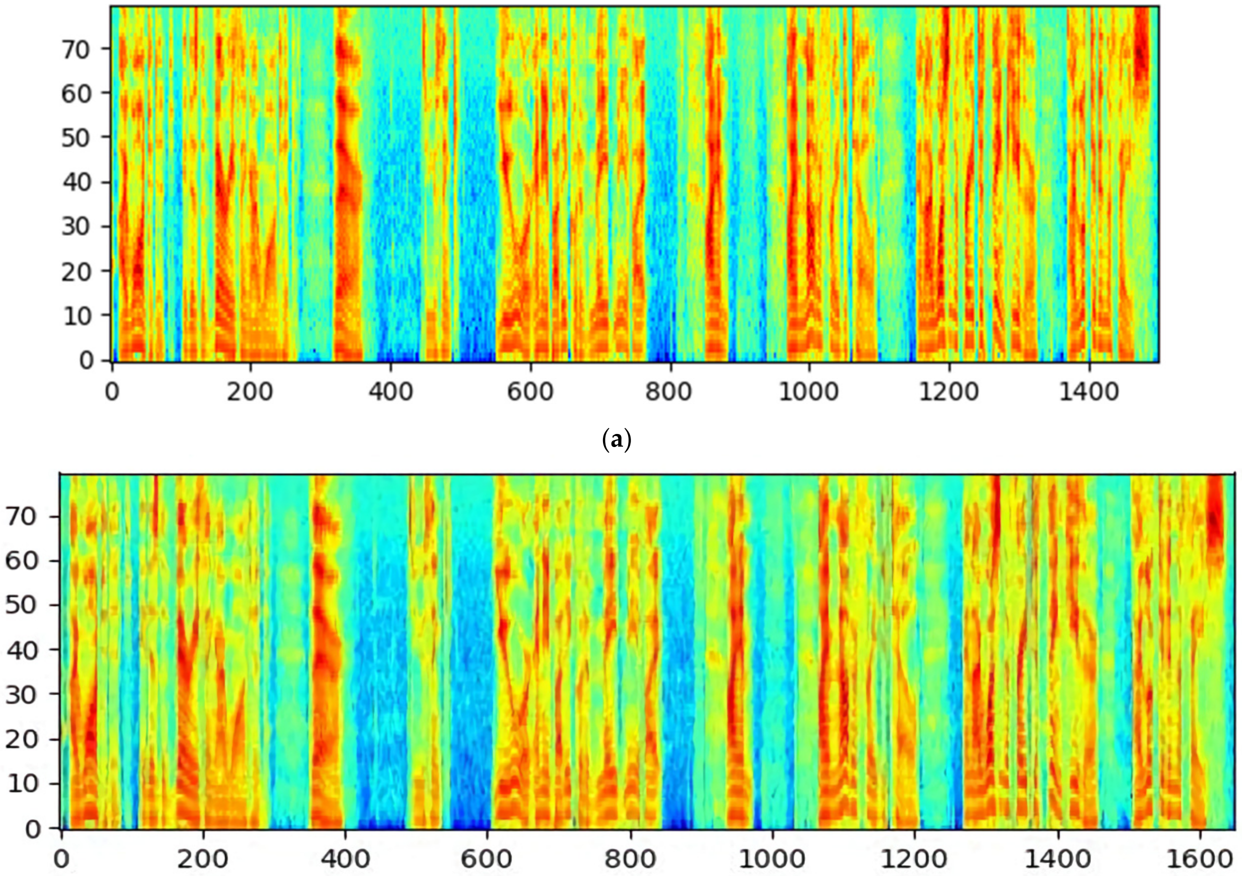
Figure 8 shows the attention alignment after learning. When speed perturbation is applied as shown in Figure 8b, the overall slope increases compared to the attention alignment of the original utterance shown in Figure 8a. However, since FrameAugment changes the speed of selected random sections of the original utterance, the slope of non-augmented sections remains unchanged, but the slope of the selected random section is changed, as shown in Figure 8c,d. The selected random section is marked with a box with red lines. We can notice that the slope increases because the frame rate is decreased in Figure 8c, and the slope decreases because the frame rate is increased in Figure 8d.



(**a**) (**b**) (**c**) (**d**)

Figure 8. Comparison of the attention alignments with the encoder feature index on x-axis and the outputs on y-axis for the same utterance. We note that speech frames are subsampled by 4:1 for encoder input of Transformer. (**a**) Original data; (**b**) speed perturbation (speed = 1.1); (**c**) FrameAugment ($s = 0.5$, $n = 300$, $p = 500$); (**d**) FrameAugment ($s = 1.5$, $n = 500$, $p = 100$).

Figure 9 shows the original spectrogram and the augmented spectrogram when FrameAugment is applied.



(**a**)

(**b**)

Figure 9. Comparison of the spectrograms with frame index on x-axis and feature index on y-axis. The length of the utterance increases due to FrameAugment. (**a**) Original spectrogram; (**b**) spectrogram with FrameAugment ($s = 1.3$, $n = 500$, $p = 100$).

Figure 10 shows the attention loss on the training data and the validation data of the LibriSpeech database using FrameAugment and other existing data augmentation methods. Figure 10a shows the loss graph for the training data. If the amount of training data for each epoch is scaled to be equal in the baseline and the speed perturbation method (i.e., 90 epochs in the baseline is scaled to 30 epochs in the speed perturbation method), the training loss curve of every data augmentation method shows convergence slower than that of the baseline.

Figure 10b shows the loss graph for the validation data, where the loss graph of every data augmentation method converged to a lower loss value than that of the baseline. SpecAugment, which augments data by masking on a random section of the spectrogram, converged to the lowest validation loss. Between the two data augmentation methods that change speed, FrameAugment converges to a lower loss value than speed perturbation. It can be expected that FrameAugment performs worse than SpecAugment but better than speed perturbation. In the experiments of the previous subsection, it was shown that FrameAugment can be constructively combined with SpecAugment and speed perturbation for the best performance.

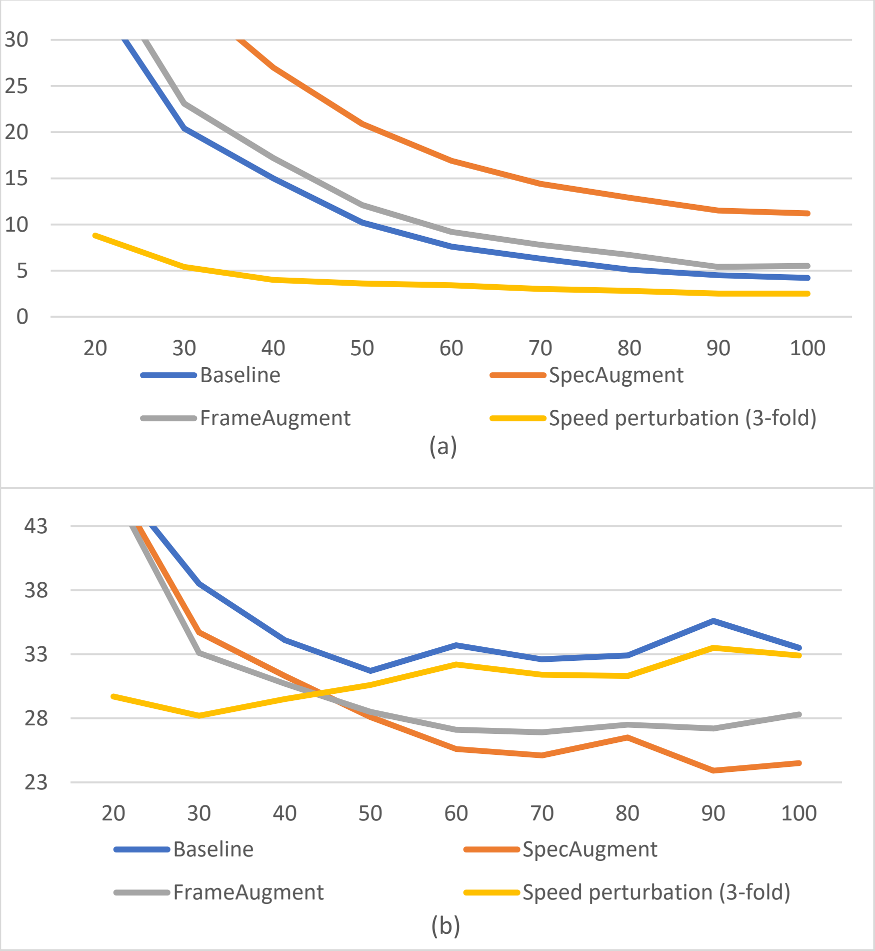


Figure 10. Attention loss $L_{Attention}$ of the training data (**a**) and validation data (**b**) of the LibriSpeech database. The horizontal axis denotes the number of epochs, and the vertical axis denotes $L_{Attention}$. We note that 3-fold speed perturbation means 3 times the original training data.

5. Conclusions

We proposed a new data augmentation method that changes the speed of some sections of speech. The proposed method was shown to achieve the best performance on average when the frame rate of 70% of utterances was changed to be 0.5–1.5. Performance was improved by about 13.6% relatively over the baseline when combined with both SpecAugment and speed perturbation in the LibriSpeech 100 h database. The proposed FrameAugment method can provide the labeled data with various speech lengths by simply changing the frame rate and can be combined with the previous data augmentation methods to achieve better results.

Further study is needed to evaluate the performance when the proposed method is used in the waveform domain or applied to other kinds of speech databases.

Author Contributions: Conceptualization, S.-S.L. and O.-W.K.; methodology, S.-S.L.; software, S.-S.L.; validation, S.-S.L.; formal analysis, S.-S.L. and O.-W.K.; investigation, S.-S.L.; resources, S.-S.L.; data curation, S.-S.L.; writing—original draft preparation, S.-S.L.; writing—review and editing, O.-W.K.; visualization, S.-S.L.; supervision, O.-W.K.; project administration, O.-W.K.; funding acquisition, O.-W.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data sharing not applicable.

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

References

1. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]
2. Xie, Q.; Dai, Z.D.; Hovy, E.; Luong, M.T.; Le, Q.V. Unsupervised data augmentation for consistency training. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 6256–6268.
3. Xie, Q.; Luong, M.T.; Hovy, E.; Le, Q.V. Self-training with Noisy Student Improves ImageNet Classification. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020.
4. Pham, H.; Dai, Z.; Xie, Q.; Le, Q.V. Meta Pseudo Labels. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Virtual, 19–25 June 2021.
5. Sohn, K.; Berthelot, D.; Carlini, N.; Zhang, Z.; Zhang, H.; Raffel, C.; Kurakin, A.; Li, C.L.; Carlini, N.; Cubuk, E.D. FixMatch: Simplifying semi-supervised learning with consistency and confidence. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 596–608.
6. Zhong, Z.; Zheng, L.; Kang, G.; Li, S.; Yang, Y. Random erasing data augmentation. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020.
7. Cubuk, E.D.; Zoph, B.; Shlens, J.; Le, Q.V. RandAugment: Practical automated data augmentation with a reduced search space. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020.
8. Yu, D.; Deng, L. *Automatic Speech Recognition—A Deep Learning Approach*; Springer: Berlin, Germany, 2016.
9. Ko, T.; Peddinti, V.; Povey, D.; Khudanpur, S. Audio augmentation for speech recognition. In Proceedings of the INTERSPEECH 2015, Dresden, Germany, 6–10 September 2015.
10. 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.
11. Song, X.; Wu, Z.; Huang, Y.; Su, D.; Meng, H. SpecSwap: A Simple Data Augmentation Method for End-to-End Speech Recognition. In Proceedings of the INTERSPEECH 2020, Shanghai, China, 25–29 October 2020.
12. Paul, D.B.; Janet, B. The design for the Wall Street Journal-based CSR corpus. In Proceedings of the Speech and Natural Language, New York, NY, USA, 23–26 February 1992.
13. Panayotov, V.; Chen, G.; Povey, D.; Khudanpur, S. LibriSpeech: An ASR corpus based on public domain audio books. In Proceedings of the ICASSP, South Brisbane, QLD, Australia, 19–24 April 2015.
14. Chan, W.; Jaitly, N.; Le, Q.V.; Vinyals, O. Listen, attend and spell. *arXiv* **2015**, arXiv:1508.01211.
15. Graves, A.; Fernández, S.; Gomez, F.; Schmidhuber, J. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In Proceedings of the 23rd International Conference on Machine Learning (ICML), Pittsburgh, PA, USA, 25–29 June 2006.
16. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 5998–6008.
17. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
18. Caruana, R. Multitask learning. *Mach. Learn.* **1997**, *28*, 41–75. [CrossRef]
19. Kim, S.; Hori, T.; Watanabe, S. Joint CTC-attention based end-to-end speech recognition using multi-task learning. In Proceedings of the 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), New Orleans, LA, USA, 5–9 March 2017.
20. Sox-Sound eXchange. Available online: http://sox.sourceforge.net/ (accessed on 17 July 2022).
21. Steffensen, J.F. *Interpolation*; Courier Corporation: North Chelmsford, MA, USA, 2006.
22. Kudo., T.; Richardson, J. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Brussels, Belgium, 31 October–4 November 2018.
23. Kudo, T. Subword regularization: Improving neural network translation models with multiple subword candidates. In Proceedings of the Association for Computational Linguistics, Melbourne, VIC, Australia, 15–20 July 2018.
24. Watanabe, S.; Hori, T.; Karita, S.; Hayashi, T.; Nishitoba, J.; Unno, Y.; Soplin, N.E.Y.; Heymann, J.; Wiesner, M.; Chen, N.; et al. ESPnet: End-to-End Speech Processing Toolkit. *arXiv* **2018**, arXiv:1804.00015.
25. 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.
26. Müller, R.; Kornblith, S.; Hinton, G.E. When does label smoothing help? *Adv. Neural Inf. Process. Syst.* **2017**, *32*, 4696–4705.