Metric Learning Approach for End-to-End Multilingual Automatic Speech Recognition Model

Akihiro Dobashi
University of Yamanashi
Kofu, Yamanashi, Japan
dobashiakihiro@alps-lab.org

Chee Siang Leow
University of Yamanashi
Kofu, Yamanashi, Japan
cheesiang_leow@alps-lab.org

Hiromitsu Nishizaki
University of Yamanashi
Kofu, Yamanashi, Japan
hnishi@yamanashi.ac.jp

***Abstract*—This study explores the application of metric learning in an end-to-end multilingual automatic speech recognition (ASR) model, employing the wav2vec 2.0 framework. In the proposed method, the E2E ASR model implements metric learning by obtaining acoustic features corresponding to character labels through forced alignment. When metric learning was applied to a six-language E2E ASR model during training, the model incorporating metric learning demonstrated a 0.7-point improvement in the character error rate (from 8.4% to 7.7%) over the baseline model, which was trained without metric learning. Additionally, the visualization of feature vectors indicated a decrease in both the variation of acoustic feature vectors for individual characters and inter-character interference, further underscoring the effectiveness of our approach.**

***Index Terms*—automatic speech recognition, multilingual ASR, metric learning, wav2vec 2.0**

I. Introduction

In recent years, research on automatic speech recognition (ASR) systems has shifted from the Hidden Markov Model (HMM)-Deep Neural Network (DNN) hybrid speech recognition framework [1] to End-to-End (E2E) types. This E2E framework is more convenient for multilingual speech recognition models. Watanabe et al. [2] proposed a single speech recognition model capable of recognizing 10 different languages using a hybrid Attention/CTC framework. However, this requires more than 1000 hours of training data, and to prevent performance degradation for languages with large amounts of data, it is necessary to build an ASR model that uses linguistic information, as shown by Toshniwal [3] et al. There have been other studies of transformer-based multi-language ASR, but many studies have used language identification [4].

To solve the problem of having a small amount of training data, a model such as wav2vec 2.0 [5] has been proposed, which uses unlabeled speech data for pre-training and labelled speech data for fine-tuning. Wav2vec 2.0 is characterized by its use of raw speech waveforms and incorporates self-supervised learning in the pre-training process. By using wav2vec 2.0's feature extractor, which has been pre-trained on a large amount of data, it is possible to learn a new task with a small amount of labelled speech data for model fine-tuning. Therefore, wav2vec 2.0 is being actively considered for use in multilingual ASR model [6].

In this paper, we introduce a novel training approach for an E2E ASR model within the metric learning framework, aiming to enhance the accuracy of the ASR model derived from wav2vec2.0. Metric learning [7] effectively classifies and clusters data by aptly defining the distance or similarity among distinct data points. Our hypothesis posits that identical characters across diverse languages should produce comparable acoustic feature vectors. Consequently, we employ metric learning to draw acoustic feature vectors closer when characters are identical and to distance them when they differ. This method seeks to augment the accuracy of a multilingual E2E ASR model.

We investigated the efficacy of metric learning in training a wav2vec2.0-based model. Experiments were conducted both with and without metric learning on a six-language E2E ASR model. Our findings suggest that the integration of metric learning led to a 0.7-point reduction in the character error rate across the six languages, in comparison to models without metric learning.

II. E2E ASR model with metric learning

A. Model Architecture

The E2E ASR model proposed in this study is based on wav2vec2.0, as shown in Fig.1. The wav2vec2.0 configuration used in this study consists of seven convolutional layers and 12 transformer encoder layers. The first convolutional layer has a kernel size of ten and a stride of five, the middle four layers have a kernel size of three and a stride of two, and the final two layers have a kernel size of two and a stride of two. The GELU function is used as the activation function in all convolutional layers. Positional Encoding is added before the input to the transformer encoder. The number of output dimensions for the transformer encoder layer is set to 768, the number of multi-heads to 12, and the number of dimensions of the feed-forward network to 3,074.

In the pre-training stage, random batches are created from the speech data of all six languages used in this study, and the model is trained according to the pre-training diagram shown in Fig. 1. In the fine-tuning stage, a fully-connected layer is added to the final layer of wav2vec 2.0, and the model is trained using the CTC loss function. In this case, as with the pre-training stage, random batches are created from the speech data in the six languages, and the model is trained according to the fine-tuning process shown in Fig. 1.

B. Applying metric learning

The summary of the metric learning proposed in this paper is shown in Fig. 2. For the integration of metric learning, forced alignment is a prerequisite. This alignment helps identify the sections of the speech that correspond to character labels. However, an inadequately trained ASR model cannot provide accurate alignment, thus necessitating sufficient model

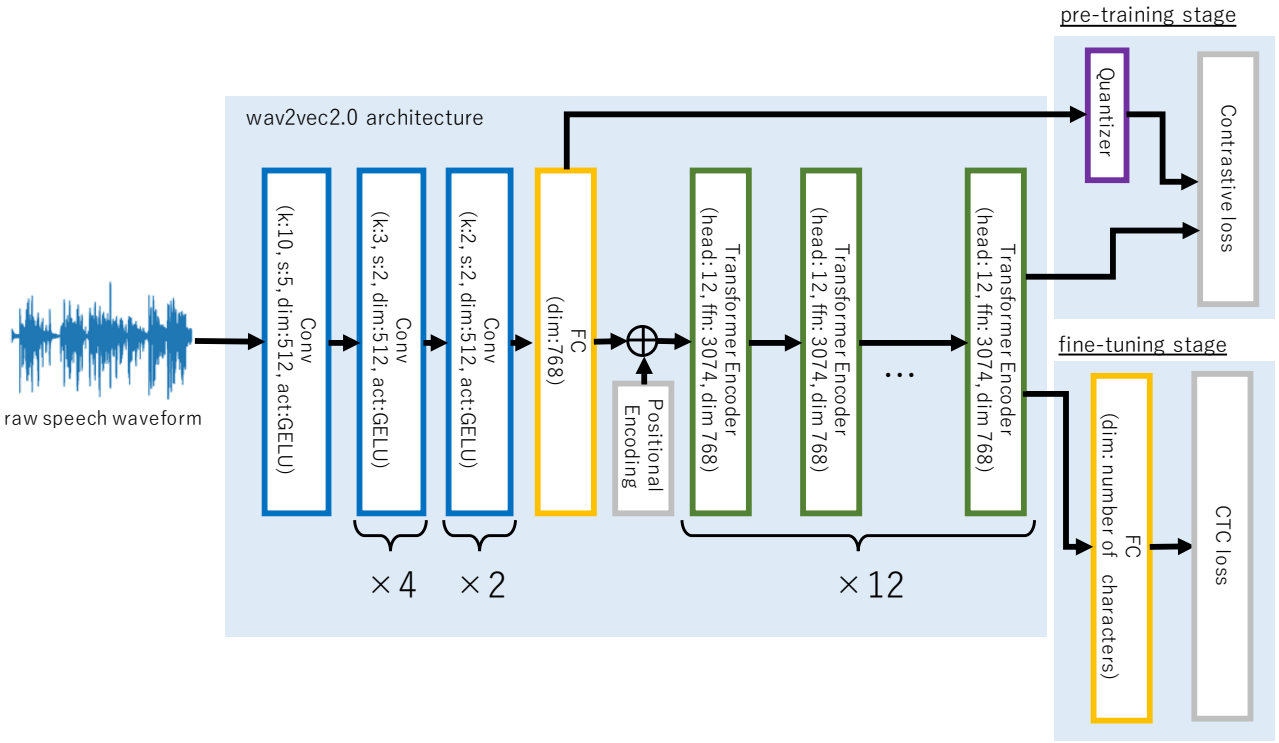

Fig. 1. The model architecture and parameters of wav2vec2.0.

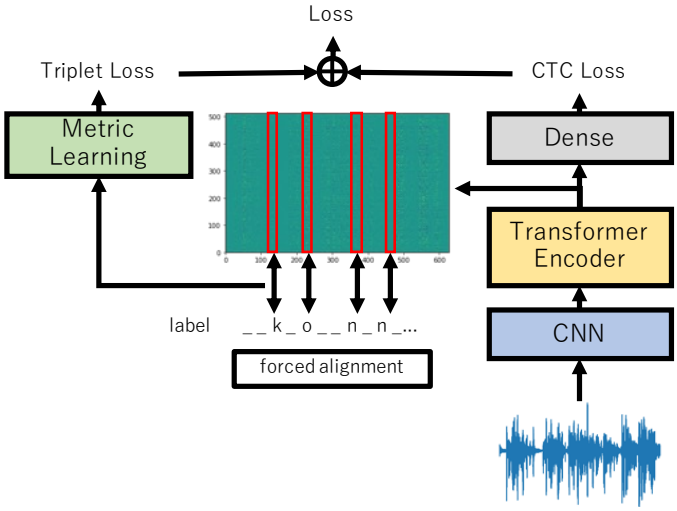

Fig. 2. ASR model training incorporating forced alignment and metric leaning.

training to obtain correct forced alignment. Once the ASR model achieves the necessary level of training for forced alignment, we proceed with fine-tuning. During fine-tuning, forced alignment helps determine the exact positions within the speech segment where each character label was spoken. We extract the features at these positions from the outputs of the transformer-encoder, treating them as the required feature vectors for character output, on which we then perform metric learning.

When trained using the CTC loss function, the ASR model frequently recognizes a single frame as a speech segment during alignment acquisition [8]. However, we posit that accurate character recognition involves not just the identified frame, but also its immediate preceding and succeeding frames. These adjacent frames contribute significantly to the precise recognition of the character. In our proposed approach, upon obtaining alignment, we treat a total of three frames (the identified frame, one frame before, and one frame after) as essential features for character output. These frames are subsequently averaged temporally to constitute a unified feature vector.

The Triplet loss function is used for metric learning and is combined with fine-tuning using the CTC loss function. This method trains the transformer encoder to produce feature vectors that are close to each other for the same character, even when that character appears across different languages.

C. Training procedure

The model based on wav2vec2.0, necessitates initial pre-training. During this phase, we employ a six-language speech corpus for the pre-training process—this same corpus is subsequently used for fine-tuning. Post pre-training, while fine-tuning is initiated, it's noteworthy that forced alignment is unattainable using only the pre-trained model. As such, we engage in fine-tuning over the six-language corpus for 500 epochs, facilitating the retrieval of forced alignment. Upon completion, the fine-tuned model undergoes additional refinement, integrating metric learning alongside the CTC loss.

The terminal training phase aspires for characters across diverse languages to converge to identical feature vectors. Given that our ASR was executed on a character-centric basis, the number of accurate labels corresponding to a singular speech utterance could surpass 200. This amplifies the complexity of universally applying metric learning across

all characters and vectors within a mini-batch. Consequently, metric learning targets merely one speech segment in a mini-batch. Since this approach confines metric learning to a single language, a specialized processing of the training data during the ultimate training phase becomes imperative. In our research, we manipulated training data to juxtapose two distinct language speeches within a singular utterance, yielding code-switched speech segments. To elaborate, pre-existing monolingual speech samples were amalgamated to generate code-switched speech. By harnessing this synthetic code-switched speech and incorporating metric learning, we endeavored to align characters from varied languages to analogous feature vectors.

The CTC loss function is defined as equation 1.

$$L_{CTC}(S) = -\sum_{\boldsymbol{x},z \in S} \log P(z|\boldsymbol{x}) \tag{1}$$

The $\boldsymbol{x}$ is the input feature, z is the correct label, and S is the union of all input features $\boldsymbol{x}$ and correct labels z used in training. In addition, the input feature $\boldsymbol{x}$ can be defined as equation 2.

$$\boldsymbol{x} = \text{Softmax}(\text{Linear}_{D \to V'}(\text{wav2vec}(\mathbf{X}))) \tag{2}$$

The $\mathbf{X}$ represents the raw waveform to be input. Softmax$(\cdot)$, Linear$_{D \to V'(\cdot)}$, and wav2vec$(\cdot)$ are the softmax function, the Linear layer that converts the output feature dimension D of wav2vec2.0 to the number of correct labels V', including blank, and all layers of wav2vec2.0.

In this paper, Triplet loss function is used for metric learning. Triplet loss function is defined as equation 3,

$$L_{Triplet} = max(\text{D}(x_{anc}, x_{pos}) - \text{D}(x_{anc}, x_{neg}) + \alpha, 0) \tag{3}$$

The D is the distance function and α is the margin value. The acoustic feature vectors corresponding to the correct answer labels identified by forced alignment are represented by x_{anc}, x_{pos}, and x_{neg}. Here, x_{anc} is the acoustic feature vector corresponding to the reference character, x_{pos} refers to the acoustic feature vector of the same character as x_{anc}, and x_{neg} refers to the acoustic feature vector of a character different from x_{anc}. In addition, since x_{anc}, x_{pos}, and x_{neg} are all features output from wav2vec 2.0, they can be expressed as equation 4.

$$x_{anc}, x_{pos}, x_{neg} \in \text{wav2vec}(\mathbf{X}) \tag{4}$$

In this paper, the margin value in the Triplet loss function was set to 0.01. When incorporating metric learning, the loss function for the model was defined as a combination of the Triplet loss and CTC loss at a ratio of 500:1. Thus, the loss function in the final training stage is expressed as equation 5.

$$L = L_{CTC} + 500 \times L_{Triplet} \tag{5}$$

III. Data sets

The data used in the experiments of this study are shown in Table I. We prepared six different languages speech corpora as follows:

- Spanish, Czech, German, and French from the GlobalPhone corpus [9]
- English from the TED-LIUM corpus [10]
- Japanese from Corpus of Spontaneous Japanese [11]

TABLE I
DATA SIZE OF TRAIN AND TEST DATA IN EACH LANGUAGE

language	train data [hours]	test data [hours]	number of characters
Czech	17.0	2.7	44
English	17.0	2.6	26
French	17.0	2.0	38
German	17.0	1.5	40
Japanese	17.0	5.1	26
Spanish	17.0	1.7	43
All lang.	102.0	15.6	66.0

In practical scenarios, the duration of training data can differ significantly across various speech corpora. For this study, we harmonized the training data duration for each language to align with the German corpus, which possesses the smallest volume at 17 hours. This alignment helped mitigate any potential degradation in ASR performance due to data volume discrepancies, ensuring a more precise assessment of our proposed method's efficacy. Moreover, we designated characters as the unit for labels. Notably, Japanese text was transcribed into Romaji, streamlining character compatibility across languages, effectively offering a phoneme-level representation. Consequently, the aggregate count of distinct characters across all languages amounted to 66.

Considering that specific models in our experiment necessitate the inclusion of code-switched speech, encompassing multiple languages within a single utterance, we devised a process for its generation. This begins by selecting a speech sample from the aforementioned corpus, succeeded by a random selection of a speech segment from a different language. These two segments are then amalgamated, yielding a composite speech sample that integrates both languages.

The character error rate (CER) is used as the measure for evaluation. CER is defined as equation 6.

$$CER = \frac{S + D + I}{N} \tag{6}$$

The S is the number of substitutions, D is the number of deletions, and I is the number of insertions. N is also the number of characters in the reference.

IV. Experiments

A. Model comparison

In the experiment, three models were trained as follows:

- **Model (1)** is trained with only CTC loss and without any code-switched speech.
- **Model (2)** is trained with only CTC loss using code-switched speech.
- **Model (3)** is trained with CTC and Triplet losses using code-switched speech.

Model (1) was trained for 1,000 epochs using speech provided by the corpus without any processing. Model (2) was also trained for 1,000 epochs but used pseudo code-switched speech according to the method described above. Model (3) underwent 500 epochs of training using the same settings as Model (1), followed by 100 epochs of fine-tuning with both the CTC and Triplet losses. The decoding method was performed using Greedy Search.

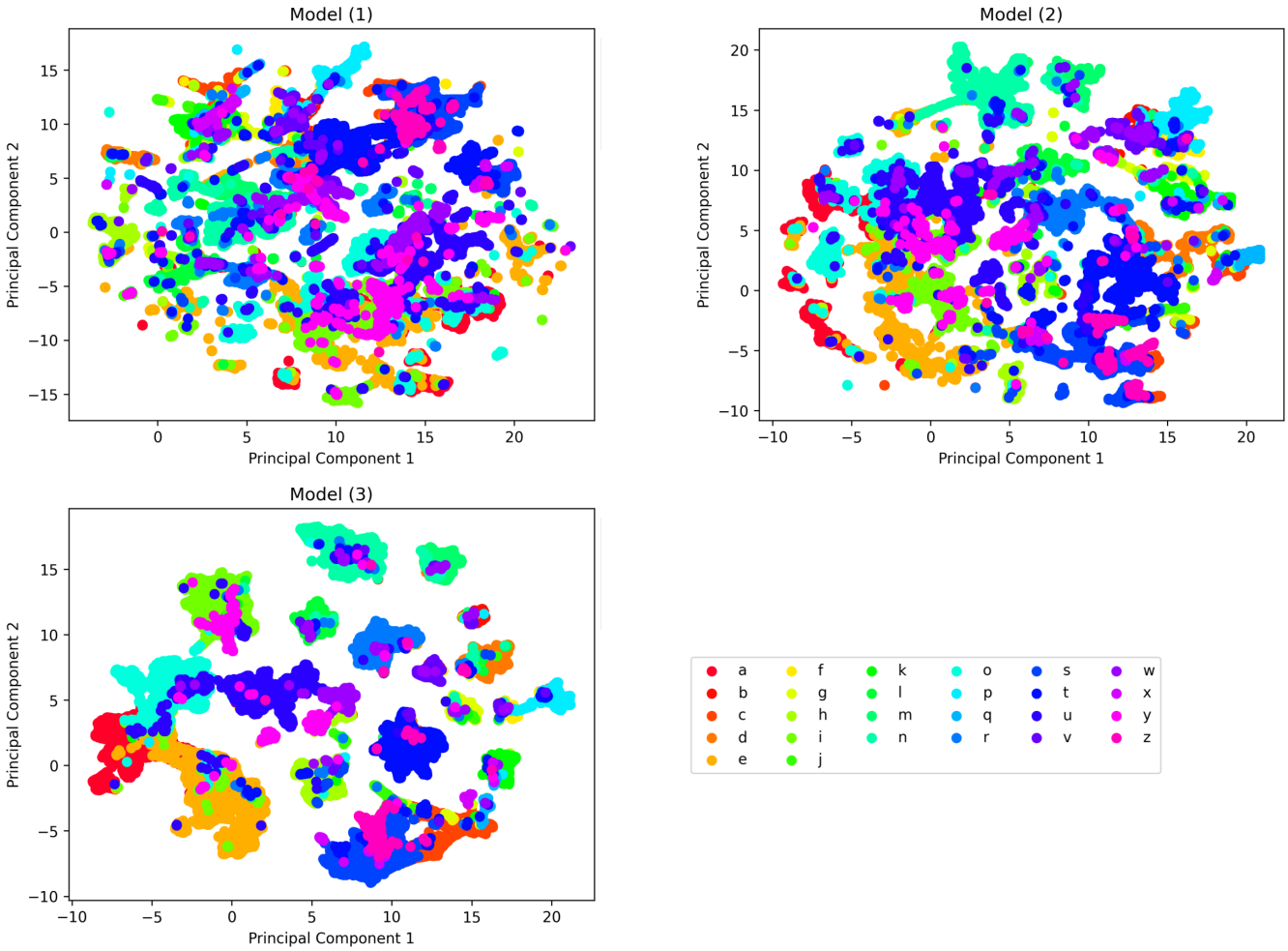

Fig. 3. Distribution of acoustic feature vectors for each character, compressed into two dimensions by UMAP. Each character is color-coded.

B. Results and discussions

Table II shows the results of the ASR experiments. These results utilize the weights of the model at the time when the best results were obtained on the evaluation data during the training of each model. Comparing Model (1) and (2), the results for Model (2) show more than a 3% improvement in CER for all languages compared to Model (1). The difference between Model (1) and (2) lies in the training data; Model (2) uses code-switched data for training. Therefore, even with the same number of epochs, the length of speech used per epoch exceeds 102 hours (six languages $\times$ 17 hours). This is believed to have led to an improvement in accuracy. Comparing Model (2) and (3), the results for Model (3) show a 0.7% improvement in CER for all languages compared to Model (2). When examining the results in terms of recognition performance by language, it is evident that the proposed method improves accuracy.

Figure 3 presents a visual representation of the acoustic feature vectors corresponding to each character. These vectors, derived through forced alignment, are dimensionally reduced to a two-dimensional space using UMAP [12]. Distinct colors denote individual characters. Though there are 66 output characters for ASR as indicated in Table I, Fig. 3 specifically highlights the 26 characters (a $\sim$ z) that are common to all languages. This selective representation is designed to evaluate the influence of metric learning on these universally shared characters. In Model (3), the distribution of feature vectors associated with each character appears markedly more compact, presenting minimal overlap with others compared to the model that excludes metric learning. This observation underscores that our metric learning implementation streamlines the training of feature vectors. It efficiently clusters identical characters while distinguishing different characters, even when

TABLE II
CERs [%] IN EACH MODEL.

Language	Model (1)	Model (2)	Model (3)
Czech	13.10	7.17	**6.21**
English	14.90	13.40	**12.3**
French	5.30	5.34	**4.99**
German	6.10	5.64	**4.61**
Japanese	15.90	9.95	**9.41**
Spanish	6.14	4.46	**4.09**
All lang.	11.70	8.41	**7.71**

spanning multiple languages.

Based on the aforementioned findings, we can conclude that the utilization of metric learning empowers the transformer-encoder to extract acoustic feature vectors that are optimized for individual characters. As a result, this optimization leads to a notable enhancement in ASR accuracy.

V. Conclusions

In this paper, we introduced a training methodology that incorporates metric learning for multilingual ASR, situated within the framework of the E2E ASR model grounded in wav2vec2.0. Metric learning requires a forced alignment process. We applied metric learning to the acoustic feature vectors associated with the characters of the correct labels. Experimental outcomes from multilingual ASR revealed that the model, integrating both the CTC loss function and metric learning, outperformed the baseline model which solely depends on the CTC loss function. Furthermore, the visualization of feature vectors confirmed that the incorporation of metric learning facilitates the extraction of apt feature vectors for diverse character classifications. This insight substantiates our assertion that metric learning significantly enhances recognition performance.

In future work, we aim to examine the efficacy of the proposed methodology by leveraging metric learning within a single-language ASR model.

References

[1] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The Kaldi Speech Recognition Toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding (ASRU 2011)*, 2011.

[2] S. Watanabe, T. Hori, and J. R. Hershey, "Language independent end-to-end architecture for joint language identification and speech recognition," in *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 265–271, 2017.

[3] S. Toshniwal, T. N. Sainath, R. J. Weiss, B. Li, P. Moreno, E. Weinstein, and K. Rao, "Multilingual speech recognition with a single end-to-end model," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4904–4908, 2018.

[4] S. Zhou, S. Xu, and B. Xu, "Multilingual end-to-end speech recognition with a single transformer on low-resource languages," 2018.

[5] A. Baevski, H. Zhou, A. Mohamed, and M. Auli, "Wav2vec 2.0: A framework for self-supervised learning of speech representations," in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, 2020.

[6] H. Yadav and S. Sitaram, "A survey of multilingual models for automatic speech recognition," in *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pp. 5071–5079, June 2022.

[7] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[8] A. Zeyer, R. Schluter, and H. Ney, "Why does ctc result in peaky behavior?," *ArXiv*, vol. abs/2105.14849, 2021.

[9] T. Schultz, "Globalphone: a multilingual speech and text database developed at karlsruhe university," in *Proceedings of ICSLP2002*, pp. 345–348, 2002.

[10] A. Rousseau, P. Deléglise, and Y. Estève, "Ted-lium: an automatic speech recognition dedicated corpus," in *Proceedings of the Conference on Language Resources and Evaluation (LREC2012)*, pp. 125–129, 2012.

[11] K. Maekawa, "Corpus of Spontaneous Japanese: Its design and evaluation," in *Proceedings of the ISCA & IEEE Workshop on Spontaneous Speech Processing and Recognition (SSPR2003)*, pp. 7–12, 2003.

[12] L. McInnes, J. Healy, N. Saul, and L. Großberger, "Umap: Uniform manifold approximation and projection," *Journal of Open Source Software*, vol. 3, no. 29, p. 861, 2018.