End-to-end Transcription of Presentations and Meetings

Masato Mimura

Abstract

This thesis presents a series of new techniques to improve the automatic transcription of real meetings and presentations with end-to-end models. End-to-end automatic speech recognition (ASR) has been intensively investigated because of its fast decoding and simplified architecture. It consists of a single neural network, unlike the hybrid HMM-based approach that uses modularized acoustic and language model components. However, it requires a considerable amount of paired data of speech and transcriptions for training, which are much more expensive to collect than unpaired speech and text resources for a target task or domain. To address this problem, we propose a framework for adapting end-to-end models using only untranscribed speech and text-only data to new domains with unseen acoustic and linguistic characteristics. Transcribing presentations and meetings is more challenging because they consist of spontaneous and long utterances. Moreover, faithfully transcribing these utterances word-by-word results in redundant and ungrammatical outputs that are not necessarily easy to read and comprehend. Therefore, we propose a method for robustly transcribing long utterances and a strategy for end-to-end generation of readable written-style texts directly from speech.

Chapter 2 gives a brief review of large vocabulary continuous speech recognition (LVCSR) and the introduction of end-to-end models.

In Chapter 3, we present an approach to cross-domain speech recognition based on acoustic feature mappings provided by a deep neural network, which is trained using non-parallel speech corpora from two domains without ground-truth labels. For training a target-domain acoustic model, we generate simulated target speech features from the labeled source domain features using a mapping. This forward mapping and the backward mapping are trained simultaneously with adversarial networks using a conventional adversarial loss and a cycle-consistency loss criterion that encourages the backward mapping to bring the translated feature back to the original as close as possible. In a highly challenging task of model adaptation only using the target-domain speech, this method achieved up to 16% relative improvements in word error rate (WER) in the evaluation using the CHiME3 real test data.

In Chapter 4, we investigate how we can leverage the latest speech synthesis technology to tailor the ASR system for a target domain by preparing only a relevant text corpus. From a set of target domain texts, we generate speech features using a sequence-to-sequence speech synthesizer. These synthesized speech features, together with real speech features from conventional speech corpora, are used to train an attention-based end-to-end ASR model. Experimental evaluations show that the proposed approach significantly improves the WER of presentation speeches of the CSJ from the baseline model trained only with real speech, although the synthetic part of the training data comes only from a single speaker voice.

Chapter 5 investigates how bidirectional attention mechanisms can be integrated to improve the performance of ASR systems. The proposed approach decodes speech from left to right and right to left, utilizing forward and backward attention vectors. The best sentence hypothesis is chosen according to the combined probabilities provided by the decoders of two directions. The proposed bidirectional decoding improved the WER by up to 13% relative for presentation speeches of the CSJ.

In Chapter 6, we propose a novel approach that outputs clean, readable text directly from a meeting speech by removing fillers and disfluent regions, substituting colloquial expressions with formal ones, inserting punctuation, recovering omitted particles, and performing other types of appropriate corrections. We formalize this approach as end-to-end generation of written-style text from speech using a single neural network. We also propose a method to guide the training of this end-to-end model using automatically generated faithful transcripts and a novel speech segmentation strategy based on online punctuation detection. An evaluation using 700-hour Japanese Parliamentary speech demonstrates that the proposed direct approach successfully generates clean transcripts for human consumption more accurately at a faster decoding speed than the conventional cascade approach. We also conduct an in-depth analysis of the edit types that professional human editors perform in creating the official written records of Japanese Parliamentary meetings and evaluate the proposed system in terms of each edit type.

Chapter 7 concludes the thesis.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Speech is a vocalized form of inter-personal communication. It provides us with a quick, flexible, inexpensive and effective way to exchange knowledge and ideas with others. It is also an advantage of oral communication that we can use and observe various types of acoustic cues such as pitch, tone, and volume. With these cues, speakers' intentions and emotions are more accurately conveyed than in written communication, and one can even engage, persuade, incite, or entertain the counterparts, which facilitates productive and creative interactions among the participants as apparent in meetings, presentations and lectures.

On the other hand, we have difficulties with information retrieval from spoken materials. For example, we cannot randomly access to a specific part of interest from a whole speech or video recording without any appropriate labels for each short segment. Therefore, in order to make the most of the knowledge and new ideas presented or produced through spoken communication, it is of great importance to not only record and store the speech data, but also build and maintain their use-friendly archives with sufficient annotations for later retrieval.

For building useful archives of speech recordings, the first fundamental step is the creation of their accurate transcriptions. Once appropriately segmented and transcribed, we can use the results for keyword search, indexing the spoken contents, hot-spot detection, generation of the time-aligned subtitle, and other advanced functionalities such as translation. With an increasing use of content platforms such as YouTube, these archiving techniques are expected to be able to transform speech to a massive source of human knowledge.

Considering that manually transcribing every new spoken material is infeasible, automatic speech recognition (ASR) is vital in this transcription process. Unlike some transactional applications such as banking and flight information service, or other recent voice assistant products, where speech is just used as spoken queries to complete simple and specific tasks, an ASR system should accept continuous speech from various topics for transcribing naturally spoken utterances. It also needs to be able to handle a large vocabulary with 100k word entries or so.

This challenging task of recognizing continuous and spontaneous speech is called *large vocabulary continuous speech recognition* (LVCSR). Among many LVCSR applications, transcription of real human-to-human interaction such as meetings and presentations is particularly difficult, because we cannot assume any strong constraint on runtime acoustic condition, topic, utterance length, and speaking style to guide a recognizer. The focus of this research is to improve the speech recognition performance for these transcription tasks by addressing a number of problems with standard LVCSR approaches.

1.1 Challenges in transcribing meetings and presentations

LVCSR models are trained with a large amount of speech and language resources in a fully data-driven way. However, when transcribing real spoken communication, an ASR system always has a chance to encounter new speakers, topics and acoustic conditions, which were not covered in the training dataset. Therefore, data scarcity in a target application domain is the major challenge for achieving a good ASR performance in real applications. This is particularly serious for end-to-end (e2e) models, because they require much more parallel training examples than hybrid systems where each component is separately built and optimized. We summarize the challenges for accurately and robustly transcribing meetings and presentations as follows.

Acoustic mismatches between training and evaluation data Statistical or deep learning-based ASR models are prone to be overfitted to particular acoustic conditions observed in the training data set, and they are unlikely to work well with new data from unknown distributions of room acoustics, noise, recording devices, speakers and speaking styles. This may be compensated by some supervised adaptation techniques, but they require a considerable amount of paired speech and labels from a target domain, which is not available in most cases.

Linguistic mismatches between training and evaluation data Every new presentation or meeting is on a specific topic with a different vocabulary and word distribution from those in the training data set, typically with unknown proper nouns or technical terms. Speaking style also affects the ASR performance. For example, spontaneous utterances include many types of spoken-language phenomena such as fillers, disfluency, and colloquial expressions. They are not observed in a read speech corpus at all.

Decoding long speech An autoregressive model is known to be superior to non-autoregressive ones in terms of speech recognition performance, and thus suitable for tasks where producing accurate transcription is of higher importance than inference speed. However, an autoregressive model has an inherent disadvantage that it can only use past predictions at each decoding step and cannot access to the predictions it will make in the later steps. This is often mentioned as *myopic bias* in autoregressive sequence prediction. This bias is more evident with longer utterances. Using more global target-side information could improve the ASR performance of an autoregressive model.

Readability of ASR outputs Since conventional ASR systems are designed to faithfully reproduce all words actually spoken in an utterance, their outputs are not necessarily easy to read and comprehend due to the existence of spoken language phenomena. For example, spontaneous utterances contain not only fillers and disfluency, but also redundant and colloquial expressions even when fluently spoken. They are often ungrammatical. Moreover, the conventional ASR systems do not produce punctuation marks at all. Consequently, a considerable amount of manual edits are required for making final texts appropriate for a written record from faithful transcripts or ASR results [1].

1.2 End-to-end modeling

This study uses e2e models for performing speech recognition and a more advanced task of generating clean transcripts directly from speech. E2e models have advantages of fast decoding and simplified architectures, which make them easy to adopt in applications, compared to complicated HMM-based hybrid systems.

However, they have a serious drawback that they always require paired speech and transcriptions for training. This means that we cannot use untranscribed speech or text resources, which are easier to collect from a target domain, for adapting an e2e model. This is contrary to ASR using a hybrid system, where we can easily expand its vocabulary by adding new word entries to the lexicon, and readily rebuild the language model component with up-to-date written materials including the latest topics.

We propose a method for creating paired training examples only from text resources that can be used for expanding the vocabulary of an e2e model and adapting it to the topics in a target domain. We also propose an adaptation method of speech features that uses only unlabeled and unpaired speech data.

1.3 Organization of this thesis

Throughout this study, we address the problems of e2e transcription of utterances that are observed in real meetings and presentations. Figure1.1 shows an overview of the contents of this thesis.

After briefly reviewing major approaches in LVCSR in Chapter 2, in order to compensate the undesirable acoustic mismatches, we propose a feature transformation framework that does not require any form of paired data based on a cycle-consistency criterion in Chapter 3.

In Chapter 4, we propose a data augmentation strategy for e2e models using text-only data. In this method, we generate synthetic paired examples that can be used for adapting an e2e ASR model using e2e text-to-speech.

In Chapter 5, we propose to improve the performance of attention-based encoder-decoder models using a dual decoder that performs both of left-to-right and right-to-left sequence predictions. They are combined together in both of the training and inference stages to incorporate the target-side global information and improve the ASR performance significantly.

In Chapter 6, we present a new speech recognition framework for generating written-style readable transcripts. This model emulates the way a human professional editor creates a written record from Parliamentary speeches by removing fillers and disfluent regions, substituting colloquial expressions with formal ones, inserting punctuation and recovering omitted particles, and performing other types of appropriate corrections.

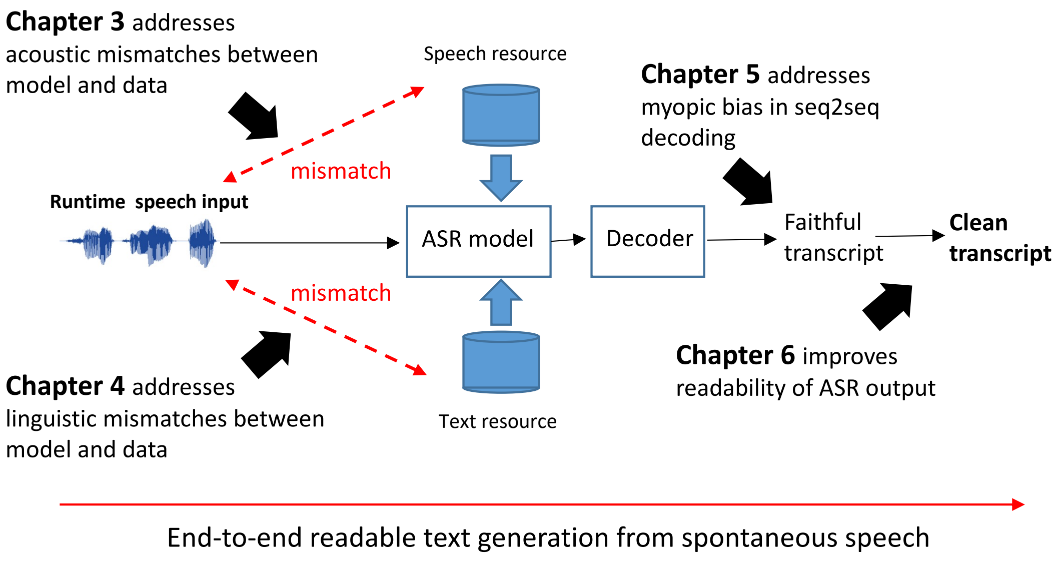

Chapter 3 addresses acoustic mismatches between model and data
Speech resource
Chapter 5 addresses myopic bias in seq2seq decoding
mismatch
Runtime speech input
ASR model
Decoder
Faithful transcript
Clean transcript
mismatch
Chapter 6 improves readability of ASR output
Chapter 4 addresses linguistic mismatches between model and data
Text resource
End-to-end readable text generation from spontaneous speech

Figure 1.1: Overview of contents of this thesis

Chapter 6 concludes the contributions of this research.

Chapter 2

Brief review on LVCSR

2.1 Statistical LVCSR system

The aim of LVCSR is to transcribe naturally spoken utterances in human-to-human communication. The major challenge here is that we cannot use task-specific constraints and the inputs are not isolated but continuous speech with diverse acoustic characteristics. This is why statistical approaches [2] using large speech and text corpora has been dominant in LVCSR. Most mainstream LVCSR systems are composed of three independent components, namely, *acoustic model*, *language model* and *pronunciation model*, all of which are built in a data-driven way.

The acoustic model learns the phonetic behavior of speakers in the training data [3]. It calculates the probabilities of acoustic observations in an utterance given a sequence of phonemes. It is a set of HMM (hidden Markov model)-based phoneme models [4][5], whose emission probability is modeled as a multiple mixture component Gaussian distribution typically with a diagonal covariance matrix (GMM-HMMs) [6]. A phoneme HMM has three hidden states or so to handle intra-phone transitions among acoustic frames, while the state-dependent GMM calculates the probability of the corresponding observation vector, usually represented as the concatenation of decorrelated Mel-frequency cepstral coefficients (MFCCs) and their delta features. A sentence-level HMM is built by simply connecting the phoneme HMMs in sequence. All parameters in a GMM-HMM system are iteratively estimated based on the Baum-Welch procedure using paired phoneme labels and acoustic feature sequences from a collection of naturally spoken utterances.

Although the use of HMMs provides us with a simple and unified framework for statistical

ASR, there are many practical difficulties to overcome for obtaining a good ASR system with sufficient accuracy. In particular, because the pronunciation of a phoneme is largely affected by its neighbors, it is vital to incorporate phonetic contexts in the training of the GMM-HMM system [3]. This is achieved by modeling phonemes with different left and right contexts using different HMM realizations. Here, in order to mitigate the data sparsity problem due to the fact that the number of training examples per each HMM state is drastically reduced when taking the phoneme contexts into account, phonetic decision tree-based clustering and parameter sharing of HMM states is performed based on statistics from the single Gaussian system [7][8].

The performance of a statistical ASR system is often not satisfactory in unknown acoustic conditions. To address the issue, many adaptation or normalization techniques were proposed to compensate the mismatches between training and evaluation data. Simply subtracting the mean vector calculated on each input utterance from the acoustic frames in the cepstrum domain (cepstral mean normalization or CMN) effectively removes multiplicative noise caused by the channel characteristics of the intermediates between the speaker and ASR system, while robustness against additive noise is obtained by normalizing the variances of cepstrum coefficients (cepstral variance normalization or CVN). Vocal tract length normalization (VTLN) [9][10] aims at ameliorating undesirable effects from speaker variations. VTLN is implemented by warping the frequency axis in the filterbank analysis with a piece-wise linear function.

More advanced adaptation methods based on statistical approaches were also proposed and widely used. Among them, maximum likelihood linear regression (MLLR) is the most popular technique for customizing the HMM parameters to the characteristics of a particular speaker [11]. In MLLR, a set of transformations are computed for reducing the mismatch between an initial model and the adaptation data from an unknown speaker. These transformations linearly modify the mean and covariance parameters in a GMM-HMM system. The MLLR parameters are estimated using an expectation-maximization (EM) procedure. Since the amount of adaptation data available for each particular speaker is limited in most cases, similar GMMs are forced to share a single regression function. The feature-space version of MLLR, namely constrained maximum likelihood linear regression or CMLLR, on the other hand, transforms the acoustic features so that each state in the HMM system is more likely to generate the adaptation data.

Discriminative training [12] is another important technique to boost the performance of an acoustic model. It aims at modifying the parameters of a GMM-HMM system so that it discriminatively assigns lower sequence-level scores to incorrect hypotheses than the correct one. In order to generate as many incorrect label examples as possible, we perform speech recognition on the training corpus using a weak language model such as unigram or bigram with a large beam width, and generate a dense lattice including many recognition paths. Many criteria and strategies for discriminative training were proposed. Examples include maximum mutual information (MMI) [12] and minimum phone error (MPE) [13]. Feature-space discriminative transformation such as fMPE [14] was also investigated and known to further improve the performance of a discriminatively trained acoustic model.

In order to decode continuous speech into a word sequence, we also use linguistic constraints in forms of a language model and pronunciation dictionary. A language model calculates the probability of a word sequence called n-gram. More specifically, an N-gram language model calculates $p(w|h)$, the probability of word w given some history h. h typically consists of a few preceding words. The parameters of the N-gram model are obtained by getting counts in a large text corpus and performing maximum likelihood estimation. Because we cannot assign any probabilities to word sequences that were not observed in the training set, N-gram smoothing techniques play an important role to avoid zero-probability. This is performed by taking some probability mass of observed N-grams and distribute it to unobserved N-grams. A number of discounting strategies were proposed including Witten-Bell, Good-Turing and Kneser-Ney discounting, and empirically Kneser-Ney discounting [15] works better than others. Domain adaptation is also important in language modeling. In most practical adaptation scenarios, it is simply done by interpolating parameters of a general model trained on a large corpus and a domain-dependent model on a limited size of text data from the target domain. The other important language component, pronunciation model or lexicon, provides a list of mappings between word entries and their pronunciations. A pronunciation entry is usually a sequence of phonemes. Each word entry may have multiple pronunciations, and the prior of each pronunciation is calculated based on its occurrence counts in the training corpus.

Once these acoustic and language components are independently built, we combine them to form a search space on which we decode an utterance into a word sequence. In this decoding, the most probable candidate is searched for using the following Bayes' rule:

$$\hat{W} = \arg\max_{W} P(W|X) = \arg\max_{W} P(X|W) \cdot P(W) \tag{2.1}$$

where X is a sequence of acoustic features and W is a candidate word sequence. In this factorized form of the posterior, the probability of the candidate word sequence $P(W)$ is calculated with the language model. The acoustic likelihood $P(X|W)$ is further factorized into the conditional distributions of acoustic features given a phonetic sequence that is calculated with the acoustic model, and the conditional probability of a phoneme sequence given a word sequence that is obtained by looking up the lexicon.

The search space for LVCSR is typically very large, which makes naive decoding strategies developed for limited-vocabulary tasks to be infeasible. Most recent systems use the weighted finite-state transducer (WFST) algorithm [16] that offers fast search and reduced runtime memory requirements for decoding. In this method, the acoustic, language and pronunciation models are separately transformed to WFST graphs, and then combined through composition, determinisation and minimization operations into one single graph with a manageable size. We perform beam search on this optimized search space.

2.2 Hybrid DNN-HMM system

Deep learning-based approaches began to be taken up in acoustic modeling for speech recognition around 2010. In one of the pioneering work [17], the deep neural network (DNN)-based LVCSR system yielded 22-28 % relative improvements in word error rate over a strong bMMI-trained baseline on a conversational speech recognition task. What is notable here is that the DNN was directly combined into the tied-state context-dependent HMM system and thus we can reuse most of the sophisticated methodologies developed for GMM-HMMs such as WFST decoding for performing LVCSR. This model, often mentioned as *hybrid DNN-HMM*, is built by replacing all of the GMM components in a GMM-HMM system with a single deep neural network (DNN) that predicts the posteriors (or scaled likelihoods) of the triphone states (senones) given an acoustic frame, The DNN part is trained using frame-level senone labels obtained by performing forced alignment using a well-trained GMM-HMM based on a cross-entropy criterion through backpropagation. The HMM parameters are simply copied from the GMM-HMM. The likelihood of a phone-state sequence required in the maximum a posteriori decoding (1.1) is calculated by dividing the DNN output vectors by the senone priors estimated on the training corpus.

In the earliest methods [18][19][17], the DNN part of a hybrid system was a simple feed-forward neural network that takes spliced acoustic features, typically consisting of 9 to 17 consecutive frames of log Mel filterbank outputs (lmfb), as input. Because a feed-forward network with a number of hidden layers is difficult to train from randomly initialized weights, pretraining methods such as unsupervised deep belief network (DBN) [20] or layer-wise discriminative training (DPT) plays an important role for avoiding an undesirable local optima. Hinton et.al summarized in [21] that the superiority of the DNNs in acoustic modeling is due to that (1) restricted Bortzmann machine used in the layer-wise pretraining of DNNs are a "product of mixtures" which is more appropriate for modeling high-dimensional data than GMMs that are a "sum of mixtures", (2) DNNs are highly nonlinear models, and (3) DNNs are good at exploiting multiple frames of input coefficients. In the subsequent research efforts, the use of many different architectures was intensively explored and it was revealed that incorporating long-term dependencies among frames is important for acoustic modeling. Among many architectures, long-short term memory (LSTM) [22] and time delay neural network (TDNN) [23]-based models were shown to be particularly effective. On the other hand, convolutional neural network (CNN)-based models are known to work well for speech recognition in adverse conditions.

Although the DNN learns to discriminate the correct senone from incorrect ones at each frame, it can further benefits from additional sentence-level discriminative training based on latices [24][25]. Many training objectives for sequence discriminative training were investigated including maximum mutual information (MMI), boosted MMI, MPE and state-level minimum Bayers risk (sMBR), among which sMBR is known to consistently work better than others in various tasks. Later, lattice-free versions of sequence discriminative training [26] such as LF-MMI or LF-sMBR were proposed and shown to not only eliminate the lattice generation process and the need for initializing the network with cross-entropy training, but also achieve better ASR performance.

Speaker or domain adaptation of the DNNs is also an important issue. Adaptation of DNNs is performed using two distinct strategies. One is based on model retraining. It uses domain-specific data for retraining the whole or part of network parameters, and thus requires a considerable amount of adaptation data and computation. The other is speaker coder-based approach. In this approach, an speaker representation vector such as i-vector obtained from a test utterance is fed into the input or intermediate layers of the DNN in addition to the standard acoustic features.

The speaker code-based one is dominant in recent research due to its computational efficiency and effectiveness.

In parallel of the dominance of the DNNs in acoustic modeling, recurrent neural networks (RNNs) gradually took over N-grams since they were firstly used for language modeling in ASR and showed impressive performance in both of perplexity and word error rate reduction in [27]. Because the RNN states are dependent on all of history words, it cannot be represented as a WFST graph. Therefore, RNN-based language models are used in the postprocessing stage for rescoring the first pass results.

2.3 End-to-end speech recognition

Both of the statistical and hybrid LVCSR systems use HMMs to deal with temporal variability of speech and combined with other components such as GMMs, DNNs and language models to decode speech. On the other hand, end-to-end (e2e) speech recognition directly maps an input feature sequence into the label sequence using a single neural network. E2e ASR eliminates the need for lexicons, language models, and complicated decoding algorithms and leads to fast decoding and simple architecture. We can use grapheme-based tokens such as characters, subwords and even words for output acoustic units in e2e ASR. E2e ASR models are categorized into two major types, namely, time-synchronous and label-synchronous models.

The first successful attempt for e2e modeling, connectionist temporal classification or CTC, was presented as early as in 2006 [28]. CTC is a loss function for training an RNN-based model that performs a direct mapping between two sequences with different lengths by introducing a special blank token for null-predictions. Although the CTC-based model was evaluated only in a small prompted speech recognition task in the original paper, it was later shown to achieve reasonable performances on major LVCSR benchmarks. CTC is particularly attractive for its fast decoding due to the non-autoregressive prediction. On the other hand, frame-level CTC predictions are conditionally independent to each other and label transitions in the output sequence are not modeled well in CTC. In RNN-Transducer [29], the label transitions are explicitly dealt with a newly introduced predictor subnetwork based on unidirectional RNN. RNN-T demonstrated the improved ASR performance over CTC. These time-synchronous models are naturally good at online streaming ASR. They are also known to robustly work in noisy conditions.

Unlike CTC and RNN-T, attention-based encoder-decoder model [30] or listen, attend spell (LAS) [31] model, directly outputs a label sequence without making any frame-level predictions. The attention-based model performs a label prediction at each decoding step dependent on the past predicted labels using unidirectional LSTM cells, as well as the accumulated encoder outputs with the attention weighting mechanism. Transformer [32] is an extension of the encoder-decoder model by replacing all recurrence calculations with self-attention. Although Transformer was originally proposed for machine translation, it was immediately shown to be also effective for various speech processing tasks. In particular, Transformer-based acoustic encoder augmented with additional convolutional component after each self-attention block, called Conformer, significantly improved the ASR performances of every type of the e2e model. Thanks to Conformer and the effective data augmentation strategies such as speed perturbation and Specaugment, the performance of the e2e models began to be competitive to or even outperform strong hybrid systems in many benchmarks.

Another significant improvement has been from self-supervised learning (SSL) of acoustic encoders. In this approach, CNN, RNN or Transformer-based encoders are trained using a large amount speech-only data based on various types of SSL tasks such as masked-input prediction. Examples include CPC [33], wave2vec-2.0 [34], and Hubert [35]. They are particularly effective in low-resource ASR scenarios with a limited amount of labeled speech.

RNN-based language models are now directly incorporated in decoding with e2e models via integration methods such as shallow fusion [36] and cold fusion, unlike in the hybrid systems. This is achieved by simply using both of the acoustic score from the ASR model and the language score from the language model to make label predictions in beam search.

In the following, we present a more detailed review on the training procedure and model architecture of three e2e methods we use in this research.

2.3.1 CTC-based model

In CTC-based speech recognition [28], we perform a mapping between two sequences with different lengths using a special token called *blank* for null-predictions. An acoustic feature sequence $\boldsymbol{X}$ is converted into a sequence of representation vectors of the same length using an acoustic encoder implemented using unidirectional or bidirectional RNN layers, or Transformer.

$$\boldsymbol{H} = \mathrm{Encoder}(\boldsymbol{X}) \tag{2.2}$$

We apply a linear transformation and softmax operation to this sequence to make frame-level predictions.

$$\boldsymbol{O} = \mathrm{Softmax}(\mathrm{Linear}(\boldsymbol{H})) \tag{2.3}$$

We can generate an expanded label sequence $\boldsymbol{\pi}$ with the same length as the input from the reference label sequence by duplicating the same label arbitrary times and inserting the blank into an arbitrary position in the label. The conditional probability of $\boldsymbol{\pi}$ is given by:

$$p(\boldsymbol{\pi}|\boldsymbol{X}) = \prod_{t=1}^{T} \boldsymbol{o}_{t,\pi_t} \tag{2.4}$$

where $\boldsymbol{o}_{t,\pi_t}$ is the corresponding element in the model output $\boldsymbol{o}_t$ to the expanded label π_t at frame t. We represent the operation of deleting blanks and duplicated labels as β. By summing up the conditional probabilities of all alignment paths that are reduced to the original label sequence Y by β, we can calculate the probability of Y.

$$p(Y|\boldsymbol{X}) = \sum_{\boldsymbol{\pi}\in\beta^{-1}(Y)}^{T} p(\boldsymbol{\pi}|\boldsymbol{X}) \tag{2.5}$$

We define the CTC loss function as the logarithm of the probability.

$$loss_{CTC}(\boldsymbol{X}, Y) = -\log(p(Y|\boldsymbol{X})) \tag{2.6}$$

As apparent from the above definition, the CTC loss strictly enforces that the input and target sequences are monotonically aligned.

2.3.2 Attention-based encoder-decoder model

In attention-based speech recognition, we model seq2seq mapping between speech and label sequences using an encoder-decoder architecture [30, 37]. This architecture has two distinct sub-networks. One is the encoder which transforms an acoustic feature sequence of length T to a sequential representation. Based on this encoded acoustic information, the other decoder sub-network predicts a label sequence whose length L is usually shorter than the input length T.

The decoder uses only a relevant portion of the encoded sequential representation for predicting a label at each time step using the attention mechanism. The encoder is implemented as multi-layer bidirectional RNN such as LSTM [22], and the decoder usually consists of a single layer of unidirectional LSTM followed by a softmax output layer.

The attention-based models are formulated as follows. The encoder transforms input acoustic features $\boldsymbol{X} = (\boldsymbol{x}_1, ..., \boldsymbol{x}_T)$ to a sequential representation $\boldsymbol{H} = (\boldsymbol{h}_1, ..., \boldsymbol{h}_T)$ that summarizes the characteristics of the input. In the following decoding step, the hidden state activation of the RNN-based decoder at the *l*-th time step is computed as:

$$\boldsymbol{r}_l = Recurrency\left(\boldsymbol{r}_{l-1}, \boldsymbol{g}_l, \boldsymbol{y}_{l-1}\right), \tag{2.7}$$

where g_l and y_{l-1} denote the "glimpse" at the *l*-th time step and the predicted label at the previous step, respectively. The glimpse g_l is a weighted sum of the encoder output sequence:

$$\boldsymbol{g}_l = \sum_t \alpha_{l,t} \boldsymbol{h}_t, \tag{2.8}$$

where $\alpha_{l,t}$ is an attention weight for $\boldsymbol{h}_t$. It is calculated as:

$$\begin{aligned} e_{l,t} &= Score(\boldsymbol{r}_{t-1}, \boldsymbol{h}_t, \boldsymbol{\alpha}_{l-1}), & (2.9) \\ \alpha_{l,t} &= \exp(e_{l,t}) / \sum_{t'=1}^{T} \exp(e_{l,t'}). & (2.10) \end{aligned}$$

There are many choices for implementation of the score function (4). In this study, we adopt the hybrid location and content-based attention mechanism [37] as follows:

$$\begin{aligned} e_{l,t} &= \boldsymbol{w}^T tanh(\boldsymbol{W}\boldsymbol{r}_{t-1} + \boldsymbol{V}\boldsymbol{h}_t + \boldsymbol{U}\boldsymbol{f}_{l,t} + \boldsymbol{b}), & (2.11) \\ \boldsymbol{f}_l &= \boldsymbol{F} * \boldsymbol{\alpha}_{l-1}, & (2.12) \end{aligned}$$

where $*$ denotes one-dimensional convolution. Using $\boldsymbol{g}_l$ and $\boldsymbol{r}_{l-1}$, the decoder predicts the next label y_l as:

$$y_l \sim Generate\left(\boldsymbol{r}_{l-1}, \boldsymbol{g}_l\right), \tag{2.13}$$

where the Generate function is implemented as:

$$\boldsymbol{R} \tanh\left(\boldsymbol{P}\boldsymbol{r}_{l-1} + \boldsymbol{Q}\boldsymbol{g}_l\right). \tag{2.14}$$

The objective for training the attention models is a cross entropy loss calculated between the predicted label sequences and the target label sequences.

Algorithm 1 ForwardBeamSearch($B, \boldsymbol{X}$)

```
1:  $F$ : set of completed label sequences
2:  $NewSeqs$ : set of label sequences at current output time
3:  $Seqs$ : set of label sequences up to the last output time
4:  $F \Leftarrow \{\phi\}$, $score(\langle \text{sos} \rangle) = 0$, $Seqs \Leftarrow \{(\langle \text{sos} \rangle)\}$
5:  $b \Leftarrow B$
6:  while $b > 0$ do
7:    $NewSeqs \Leftarrow \{\phi\}$
8:    for sequence $\boldsymbol{s} \in Seqs$ do
9:      $Y \Leftarrow$ The $b$ best words in terms of $p(y|\boldsymbol{s}, \boldsymbol{X})$
10:     for word $y \in Y$ do
11:       $\boldsymbol{s}^+ \Leftarrow concat(\boldsymbol{s}, y)$
12:       $score(\boldsymbol{s}^+) = score(\boldsymbol{s}) + \log(p(y|\boldsymbol{s}, \boldsymbol{X}))$
13:       if $y = \langle \text{eos} \rangle$ then
14:         add $\boldsymbol{s}^+$ to $F$
15:         $b \Leftarrow b - 1$
16:       else
17:         add $\boldsymbol{s}^+$ to $NewSeqs$
18:       end if
19:     end for
20:   end for
21:   $Seqs \Leftarrow$ The $b$ best sequences in $NewSeqs$ in terms of $score(\boldsymbol{s}^+)$
22: end while
23: return set of sentence candidates $F$
```

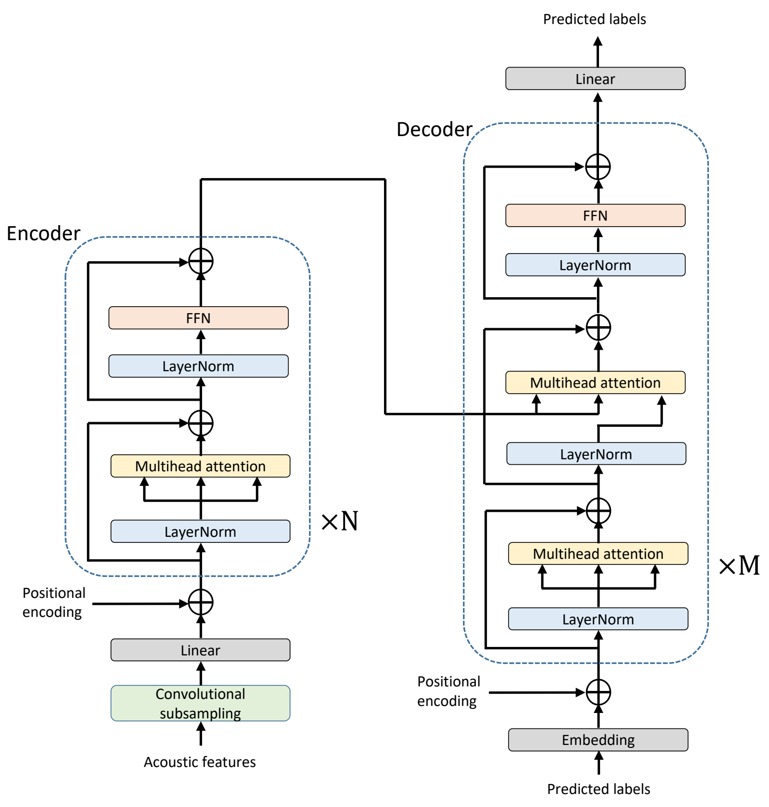


Figure 2.1: Transformer-based ASR model

A runtime decoding algorithm for word-level attention-based models is presented in Algorithm 1. This algorithm returns the B-best sentence candidates using a decreasing beam width initialized with B. It is simple since we do not need to incorporate external dictionaries or language models. ⟨sos⟩ and ⟨eos⟩ are special symbols for the start and end of a sentence. The posterior probability of a word at each decoding step $p(y|\boldsymbol{s}, \boldsymbol{X})$ on line 9 and 12 is calculated using formulas from (1) to (8). After performing Algorithm 1, we rescore each sentence candidate $\boldsymbol{s}$ in F using an insertion penalty λ as follows:

$$rescore(\boldsymbol{s}) = score(\boldsymbol{s}) - \lambda|\boldsymbol{s}|, \tag{2.15}$$

where $|\boldsymbol{s}|$ is the length of sequence $\boldsymbol{s}$, and $score(\boldsymbol{s})$ is the value calculated on line 12 of Algorithm 1. We output the sentence with the largest $rescore(\boldsymbol{s})$.

2.3.3 Transformer

Transformer [32] is a variant of sequence-to-seuqnce models consisting of distinct encoder and decoder subnetworks. Transformer was originally proposed for machine translation, but later shown to be also effective in various speech processing tasks [38][39][40]. We depict the architecture of a Transformer-based ASR model in figure2.1.

The output of each Transformer layer is calculated using the multihead attention mechanism as follows:

$$\mathrm{Multihead}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \mathrm{Concat}(head_1, head_2, ..., head_h)\boldsymbol{W}^o \tag{2.16}$$

$$head_i = \mathrm{Attention}(\boldsymbol{Q}\boldsymbol{W}_i^Q, \boldsymbol{K}\boldsymbol{W}_i^K, \boldsymbol{V}\boldsymbol{W}_i^V) \tag{2.17}$$

$$\mathrm{Attention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \mathrm{Softmax}(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d_k}})\boldsymbol{V} \tag{2.18}$$

where the dimensionality of the output and the number of attention heads are defined as d_{model} and h, respectively, and $\boldsymbol{Q} \in \mathbb{R}^{t_q \times d_q}$, $\boldsymbol{K} \in \mathbb{R}^{t_k \times d_k}$, $\boldsymbol{V} \in \mathbb{R}^{t_v \times d_v}$, $\boldsymbol{W}_i^Q \in \mathbb{R}^{d_{model} \times d_q}$, $\boldsymbol{W}_i^K \in \mathbb{R}^{d_{model} \times d_k}$, $\boldsymbol{W}_i^V \in \mathbb{R}^{d_{model} \times d_v}$, $\boldsymbol{W}^O \in \mathbb{R}^{hd_v \times d_{model}}$. Here, we usually assume $t_k = t_v$ and $d_q = d_k = d_v = d_{model}/h$. Using this multihead attention mechanism, the output sequence of each encoder layer $\boldsymbol{X}_{enc}^n$ is given by:

$$\boldsymbol{A}_{enc}^n = \mathrm{LayerNorm}(\boldsymbol{X}_{enc}^{n-1}) \tag{2.19}$$

$$\boldsymbol{B}_{enc}^n = \boldsymbol{X}_{enc}^{n-1} + \mathrm{Multihead}(\boldsymbol{A}_{enc}^n, \boldsymbol{A}_{enc}^n, \boldsymbol{A}_{enc}^n) \tag{2.20}$$

$$\boldsymbol{C}_{enc}^n = \mathrm{LayerNorm}(\boldsymbol{B}_{enc}^n) \tag{2.21}$$

$$\boldsymbol{X}_{enc}^n = \boldsymbol{B}_{enc}^n + \mathrm{FFN}(\boldsymbol{C}_{enc}^n) \tag{2.22}$$

where $\boldsymbol{X}_{enc}^0 = \boldsymbol{X} + \boldsymbol{P}$, using a sinusoidal positional encoding $\boldsymbol{P}$.

On the other hand, the output of each decoder layer is calculated using both of the output of the previous decoder layer $\boldsymbol{X}_{dec}^{m-1}$ and the encoder output $\boldsymbol{X}_{enc}^N$. Note that we define $\boldsymbol{X}_{dec}^0 = \mathrm{Embedding}(Y) + \boldsymbol{P}$.

$$\boldsymbol{A}^{m}_{dec} = \mathrm{LayerNorm}(\boldsymbol{X}^{m-1}_{dec}) \tag{2.23}$$

$$\boldsymbol{B}^{m}_{dec} = \boldsymbol{A}^{m-1}_{dec} + \mathrm{Multihead}(\boldsymbol{A}^{m}_{dec}, \boldsymbol{A}^{m}_{dec}, \boldsymbol{A}^{m}_{dec}) \tag{2.24}$$

$$\boldsymbol{C}^{m}_{dec} = \mathrm{LayerNorm}(\boldsymbol{B}^{m}_{dec}) \tag{2.25}$$

$$\boldsymbol{D}^{m}_{dec} = \boldsymbol{B}^{m}_{dec} + \mathrm{Multihead}(\boldsymbol{C}^{m}_{dec}, \boldsymbol{X}^{N}_{enc}, \boldsymbol{X}^{N}_{enc}) \tag{2.26}$$

$$\boldsymbol{E}^{m}_{dec} = \mathrm{LayerNorm}(\boldsymbol{D}^{m}_{dec}) \tag{2.27}$$

$$\boldsymbol{X}^{m}_{dec} = \boldsymbol{D}^{m}_{dec} + \mathrm{FFN}(\boldsymbol{E}^{m}_{dec}) \tag{2.28}$$

We perform label prediction at each decoding step using the output of the final decoder layer $\boldsymbol{X}^{M}_{dec}$ as:

$$\boldsymbol{O} = \mathrm{Softmax}(\mathrm{Linear}(\boldsymbol{X}^{M}_{dec})) \tag{2.29}$$

We define the loss function as cross-entropy between this model prediction and the reference label sequence.

$$loss_{Transformer}(\boldsymbol{X}, Y) = \sum_{n=1}^{N} \mathrm{onehot}(y_n) \log(\boldsymbol{o}_n) \tag{2.30}$$

Chapter 3

Cross-domain speech recognition using nonparallel corpora with cycle-consistent adversarial networks

There are a number of potential needs for domain adaptation of acoustic models in ASR, because the performance of ASR systems is known to be drastically degraded in unkown acoustic conditions as we mentioned in Section 1. One example is a noisy speech recognition task. ASR in noisy conditions was conventionally addressed by multi-condition training of acoustic models using simulated noisy corpora, which is artificially generated by convolving room impulse responses and adding noise to clean corpora. Another approach is acoustic feature enhancement in frontend using denoising autoencoders [41][42][43][44][45], where mappings from noisy features to enhanced features are learned using paired examples between clean and simulated noisy corpora. The problem with these methods is that the mixing process of speech and noise in real noisy conditions may have a highly nonlinear nature, and the simulated data generated using linear transformations described above has a very different characteristics from real noisy data, which can limit the performances of the methods based on simulated data. In fact, discrepancies between the recognition performances for simulated and real noisy test data have been reported in the literature [46][47]. Generating simulated data also requires a considerable cost for carefully recording room impulse responses and noise backgrounds in the target conditions which can sometimes lead to an infringement of privacy. On the other hand, annotating real noisy corpus is highly expensive, and paired examples between clean and real noisy data can not be generated in principle, as in most of other adaptation problems such as speaker or speaking style adaptation.

In this chapter, we propose a novel approach to cross-domain speech recognition requiring no corresponding examples between source and target domains and no labels for the target domain corpus. In the proposed method, acoustic models are trained using "fake" target domain features translated from source domain features using a complex nonlinear mapping provided by a variant of generative adversarial networks (GANs) [48]. This network is trained without supervision using source and target domain corpora. The method does not require initial speech recognition results which are crucial in typical retraining approaches for unsupervised acoustic model adaptation. Two notable design choices are incorporated into the network including the cycle consistency loss criterion [49] in the training to guide the network to retain useful information for speech recognition in the translated features. In the experiment, we also demonstrate that "fake" source domain features generated by an inverse mapping from target to source domain can contribute to improve the speech recognition performance.

After we introduce our proposed method for cross-domain speech recognition in Section 2, the detail of the network architectures and the training procedure is explained in Section 3. Experimental evaluations are presented in Section 4 before the conclusion in Section 5.

3.1 Acoustic model adaptation with generative adversarial networks (GANs)

The problem we address in this chapter is summarized as follows. We have two kinds of data which belong to two distinct domains, namely, "source" and "target" domains. For the source domain data, we have manual transcriptions which can be used for training acoustic models, but we do not for the target domain. We also do not have paired examples between these two domains. Our objective is to improve ASR performance for test data which belongs to the target domain under these constraints.

This is a very common situation we encounter frequently. In the following parts of this section, we particularly take an "noise-robust ASR" example where the source domain is clean

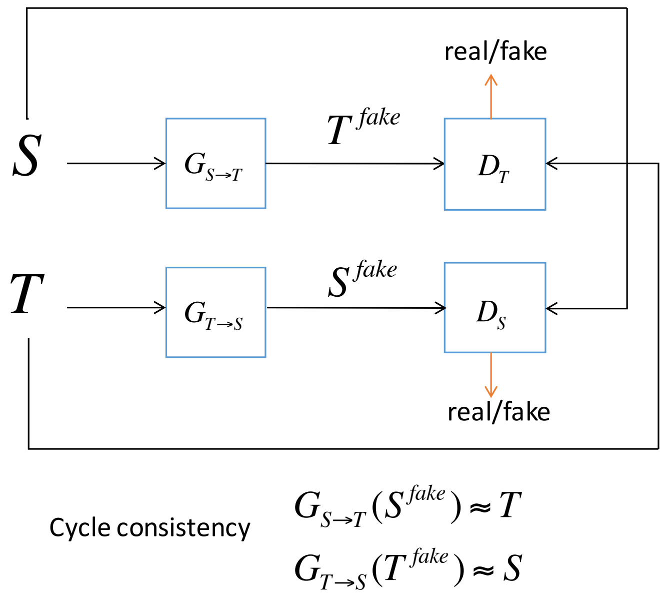


Figure 3.1: Original cycle GAN architecture for image-to-image translation

speech and the target domain is noisy speech in explaining our proposed method for understandability. Our approach is to train acoustic models using "fake" noisy features translated from clean features for which we have transcriptions. The most important issue here is how to generate a realistic "noisy" version of clean features in the absence of paired examples between two domains. This is a much more difficult setting than in conventional denoising autoencoder approaches ([50][51][41][42][43][44][45]).

We propose to use the concept of generative adversarial networks (GANs) [48] and cycle consistency adversarial networks (cycle GANs) [49], which recently yielded impressive results in the image processing area, for generating translated data. Moreover, we introduce an enhancement in the architecture of GANs in order to achieve desirable characteristics for ASR in translated features.

3.1.1 Generative adversarial networks for domain translation

GAN is a framework for estimating generative models via an adversarial process, in which two models G and D are simultaneously trained. G is a generative model that captures the data distribution, and D is a discriminative model that estimates the probability that a sample came from the training data rather than G. While the original GANs generate data from latent variables, we consider here a network $G_{S\to T}$ which transforms speech from a domain S (e.g. clean speech) to some other domain T (e.g. noisy speech). This generator network $G_{S\to T}$ is trained such that the distribution $p_{tgt}(\boldsymbol{t})$ of speech features t in T is indistinguishable from the distribution of "fake" target domain speech features $G_{S\to T}(\boldsymbol{s})$, where s is subject to the source domain distribution $p_{src}(\boldsymbol{s})$. In the GAN framework, this is achieved by optimizing the following minimax criterion:

$$G^*_{S\to T} = \arg \min_{G_{S\to T}} \max_{D_T} \mathcal{L}_{\text{GAN}}(G_{S\to T}, D_T), \tag{3.1}$$

where an *adversarial objective* $\mathcal{L}_{\text{GAN}}(G_{S\to T}, D_T)$ is defined as:

$$\begin{aligned} \mathcal{L}_{\text{GAN}}(G_{S\to T}, D_T) = \mathbb{E}_{\boldsymbol{t}\sim p_{tgt}(\boldsymbol{t})}[\log D_T(\boldsymbol{t})] + \\ \mathbb{E}_{\boldsymbol{s}\sim p_{src}(\boldsymbol{s})}[\log(1 - D_T(G_{S\to T}(\boldsymbol{s})))], \end{aligned} \tag{3.2}$$

and D_T is a discriminator network with its output representing the probability that the input comes from T. By the optimization criterion (1), D_T is trained to maximize $D_T(t)$ for $t \in T$ and minimize $D_T(G_{S\to T}(s))$ for fake data $G_{S\to T}(s)$ generated from $s \in S$, while $G_{S\to T}$ is trained to maximize $D_T(G_{S\to T}(s))$.

3.1.2 Cycle consistency loss

GANs may provide us with a powerful way to translate data across domains without parallel examples, but they are too under-constrained for keeping discriminative information required in ASR. For example, information such as formant trajectories needs to be preserved after domain translation, but the adversarial loss (2) may not enough for it.

Therefore, we choose to train not only a source to target mapping $G_{S\to T}$, but also a target to source mapping $G_{T\to S}$ in a consistent way by introducing a constraint that these mappings should be "cycle consistent" [49], namely, the data translated by $G_{S\to T}$ is mapped back by $G_{T\to S}$ to a source domain feature as close as possible to the original feature, and vice versa. Thus, we expect that the information for reconstructing the input data in either domain is kept in the domain-transformed data. We can incentivize this behavior using a *cycle consistency loss*:

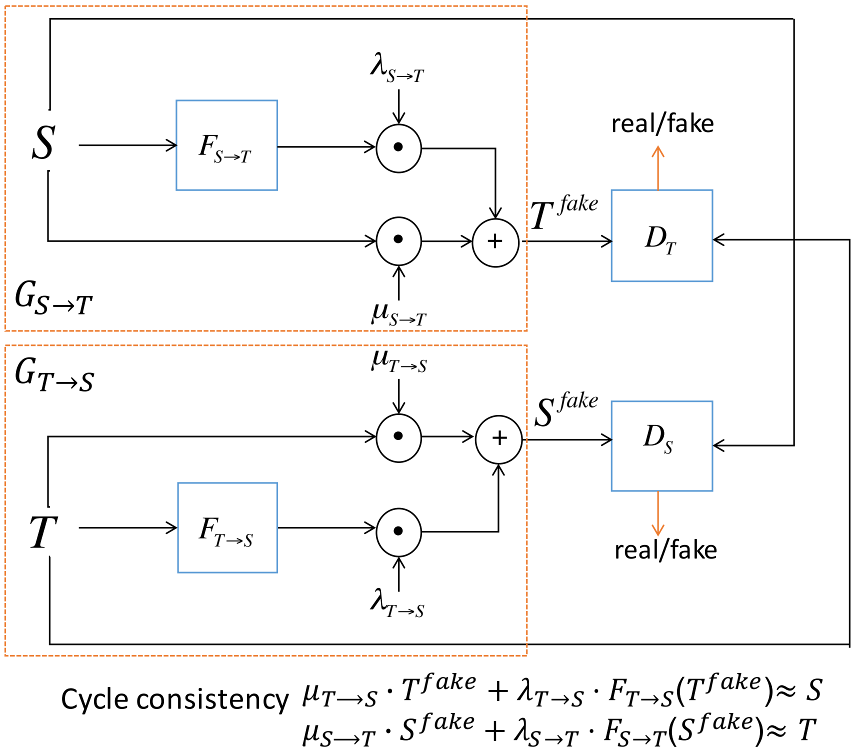

S
$F_{S\to T}$
$\lambda_{S\to T}$
$\mu_{S\to T}$
$G_{S\to T}$
T^{fake}
D_T
real/fake
$G_{T\to S}$
$\mu_{T\to S}$
T
$F_{T\to S}$
$\lambda_{T\to S}$
S^{fake}
D_S
real/fake
Cycle consistency $\mu_{T\to S}\cdot T^{fake}+\lambda_{T\to S}\cdot F_{T\to S}(T^{fake})\approx S$
$\mu_{S\to T}\cdot S^{fake}+\lambda_{S\to T}\cdot F_{S\to T}(S^{fake})\approx T$

Figure 3.2: Proposed cycle GAN-based architecture for acoustic feature transformation

$$\begin{aligned}\mathcal{L}_{\text{cyc}}(G_{S\to T}, G_{T\to S}) &= \\ \mathbb{E}_{\boldsymbol{s}\sim p_{src}(\boldsymbol{s})}&[\|G_{T\to S}(G_{S\to T}(\boldsymbol{s}))-\boldsymbol{s}\|_1] \\ +\mathbb{E}_{\boldsymbol{t}\sim p_{tgt}(\boldsymbol{t})}&[\|G_{S\to T}(G_{T\to S}(\boldsymbol{t}))-\boldsymbol{t}\|_1]. \end{aligned} \tag{3.3}$$

By paring this cycle consistency loss with the standard adversarial loss, we encourage $G_{S\to T}(G_{T\to S}(\boldsymbol{t})) \approx \boldsymbol{t}$ and $G_{T\to S}(G_{S\to T}(\boldsymbol{s})) \approx \boldsymbol{s}$. The structure of GANs with the cycle consistency loss is depicted in Fig. 3.1.

The full objective for training $G_{S\to T}$, $G_{T\to S}$, D_S and D_T is:

$$\begin{aligned}\mathcal{L}(G_{S\to T}, G_{T\to S}, D_S, D_T) = \mathcal{L}_{\text{GAN}}(G_{S\to T}, D_T) \\ +\mathcal{L}_{\text{GAN}}(G_{T\to S}, D_S) \\ +\alpha\mathcal{L}_{\text{cyc}}(G_{S\to T}, G_{T\to S}). \end{aligned} \tag{3.4}$$

3.1.3 Modified network architecture for acoustic feature transformation

We also consider to add more structure to our network architecture for further improving the quality of the transformed data.

We build two distinct paths in each of $G_{S\to T}$ and $G_{T\to S}$, which will be summed together to generate transformed data (Fig. 3.2). The first path is basically an identity mapping, which explicitly guarantees that the detailed structure in the input data will be preserved after the domain translation. The second has a generative network $F_{S\to T}$ or $F_{T\to S}$ which has a capacity to learn a complex nonlinear mapping like generators in standard GANs. With the existence of the first identity mapping path, the generative networks in the second path are enforced to learn devotedly the "difference" between two domains. Submapping-wise scaling factors $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$ are introduced for adjusting the intensity of each component before summation.

Now the mappings $G_{S\to T}$ and $G_{T\to S}$ are reformulated as:

$$G_{S\to T}(\boldsymbol{s}) = \boldsymbol{\lambda}_{S\to T} \odot F_{S\to T}(\boldsymbol{s}) + \boldsymbol{\mu}_{S\to T} \odot \boldsymbol{s}, \tag{3.5}$$

$$G_{T\to S}(\boldsymbol{t}) = \boldsymbol{\lambda}_{T\to S} \odot F_{T\to S}(\boldsymbol{t}) + \boldsymbol{\mu}_{T\to S} \odot \boldsymbol{t}, \tag{3.6}$$

where $\odot$ means element-wise multiplication.

3.2 Implementation

3.2.1 Network architecture

Following the description in the implementation part of [49], the architecture for the generative networks $F_{S\to T}$ and $F_{T\to S}$ (Fig. 3.3) is adapted from Johnson et al. [52]. A feature map of size 40x11x1 consisting of eleven frames of 40-channel log Mel-scale filterbank (lmfb) outputs is used as input to the networks. Each generative network has two subnetworks for downsampling and upsamping. The downsampling subnetwork consists of three convolutional layers and nine residual blocks [53]. Each residual network is composed of two stride-1 convolutions and a residual connection which bypasses them. On the other hand, the upsampling part has two deconvolutional layers followed by one stride-1 convolution. The filter and stride size in each convolution layer are depicted in Fig. 3.3. For example, (conv,3x3,1x1) means a convolution layer with a filter of size 3x3 and a stride of size 1x1. Note that we chose a smaller size

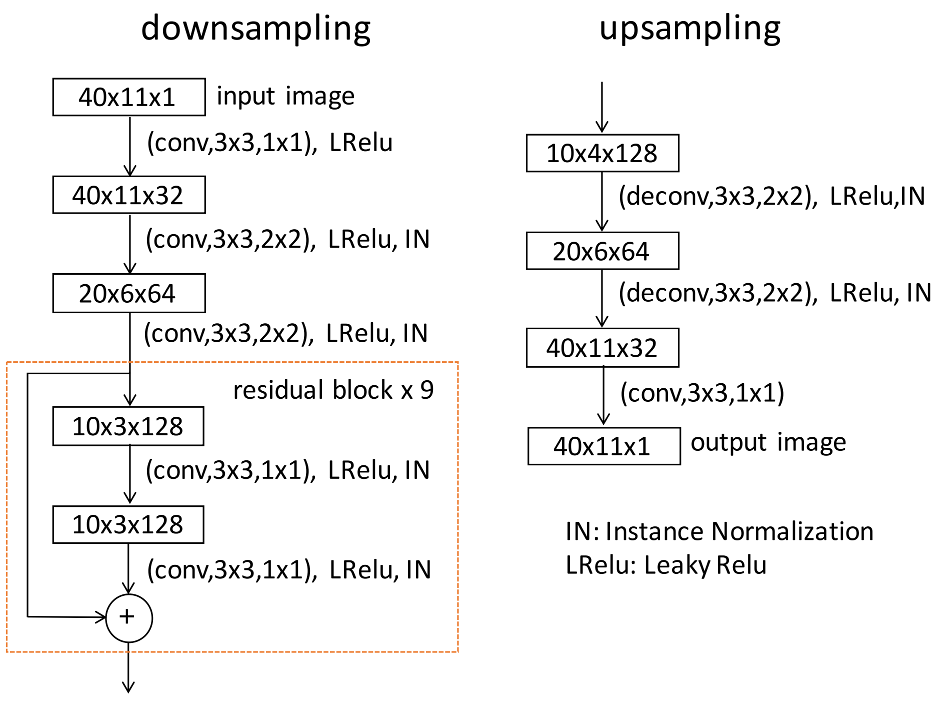


Figure 3.3: Network architecture for $F_{S \to T}$ and $F_{T \to S}$

for convolutional filters than in [49], because the size of our input images composed of lmfb features is much smaller than those in typical image processing applications. We used instance normalization [54] in layers specified in Fig. 3.3 before applying nonlinearities. Leaky ReLU nonlinearities with a slope of 0.2 are used in all layers with the exception of the output layer, which uses an identity function.

The architecture for the discriminators D_S and D_T is depicted in Fig. 3.4. The network has two convolutional layers, followed by three fully-connected layers. While Leaky ReLU nonlinearities are also used in the discriminators, we did not apply instance normalization here as suggested in [55].

3.2.2 Training procedure

We used Wasserstein GANs (WGANs) [56] for building our generative networks instead of standard GANs to stabilize our model training procedure and avoid training problems inherent in GANs such as model collapse. Gradient penalties recently proposed in [55] are also used in training WGANs instead of the weight clipping technique [56] to enforce the Lipshitz constraint.

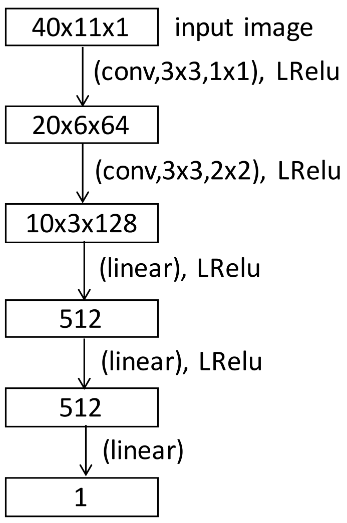


Figure 3.4: Network architecture for D_S and D_T

Accordingly, the full objective (4) is modified as:

$$\begin{aligned}\mathcal{L}(G_{S\to T}, G_{T\to S}, D_S, D_T) &= \mathcal{L}_{\text{WGAN}}(G_{S\to T}, D_T)\\ &+\mathcal{L}_{\text{WGAN}}(G_{T\to S}, D_S)\\ &+\alpha\mathcal{L}_{\text{cyc}}(G_{S\to T}, G_{T\to S}),\end{aligned} \tag{3.7}$$

where $\mathcal{L}_{\text{WGAN}}(G_{S\to T}, D_T)$ is:

$$\begin{aligned}\mathcal{L}_{\text{WGAN}}(G_{S\to T}, D_T) &= \mathbb{E}_{\boldsymbol{t}\sim p_{tgt}(\boldsymbol{t})}[D_T(\boldsymbol{t})]\\ &-\mathbb{E}_{\boldsymbol{s}\sim p_{src}(\boldsymbol{s})}[D_T(G_{S\to T}(\boldsymbol{s}))]\\ &-\beta\mathbb{E}_{\hat{\boldsymbol{t}}\sim p_{tgt}(\hat{\boldsymbol{t}})}[(\|\Delta_{\hat{\boldsymbol{t}}}D_T(\hat{\boldsymbol{t}})\|_2-1)^2]\\ &-\beta\mathbb{E}_{\hat{\boldsymbol{s}}\sim p_{src}(\hat{\boldsymbol{s}})}[(\|\Delta_{\hat{\boldsymbol{s}}}D_S(\hat{\boldsymbol{s}})\|_2-1)^2],\end{aligned} \tag{3.8}$$

where, for example, $\mathbb{E}_{\hat{\boldsymbol{t}}\sim p_{tgt}(\hat{\boldsymbol{t}})}[(\|\Delta_{\hat{\boldsymbol{t}}}D_T(\hat{\boldsymbol{t}})\|_2-1)^2]$ is the gradient penalty for critic[1] D_T. We defined $\hat{\boldsymbol{t}}$ as $\hat{\boldsymbol{t}} = a\boldsymbol{t} + (1-a)G_{S\to T}(\boldsymbol{s})$ using a random variable $a \sim U(0,1)$, $\boldsymbol{s} \sim p_{src}(\boldsymbol{s})$ and $\boldsymbol{t} \sim p_{tgt}(\boldsymbol{t})$, as suggested in [55]. More detailed explanations on WGANs and gradient penalties, which are out of scope of this research, are found in [56] and [55].

Following the recipes in [56] and [55], we update critics D_S and D_T n_{critic} times before updating $G_{S\to T}$ and $G_{T\to S}$ for each minibatch iteration. Note that while the critics are trained to mini-

[1]When we use WGANs, we call the networks D_S and D_T "critics", because they actually don't discriminate anything.

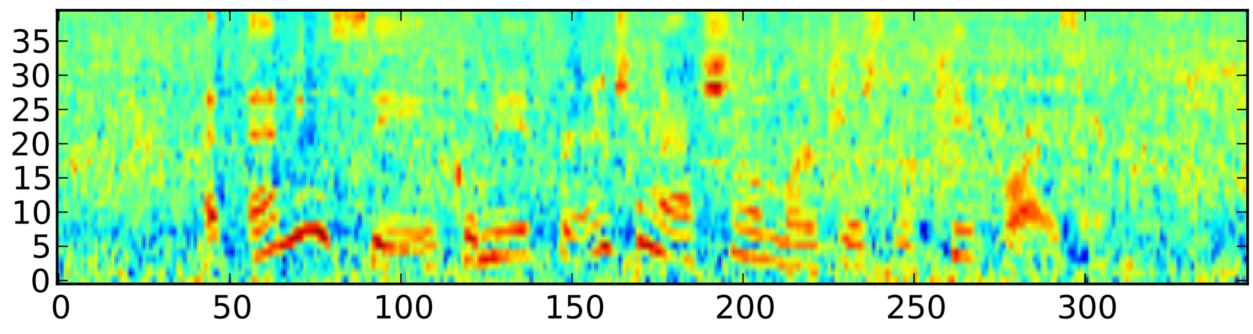
mize $-\mathcal{L}(G_{S\to T}, G_{T\to S}, D_S, D_T)$, the generators are trained to minimize $\mathcal{L}(G_{S\to T}, G_{T\to S}, D_S, D_T)$. We set n_{critic} to be 4 in all our experiments. We used the Adam optimizer [57] with a minibatch size of 256 for each of source and target domain data. All network parameters are initialized with random values with the exception of $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$, which were initialized with 1, and trained with a learning rate of 0.0001 for 20 epochs. We set α in (7) to be 10, and β in (8) to be 10.

3.3 Experimental evaluations

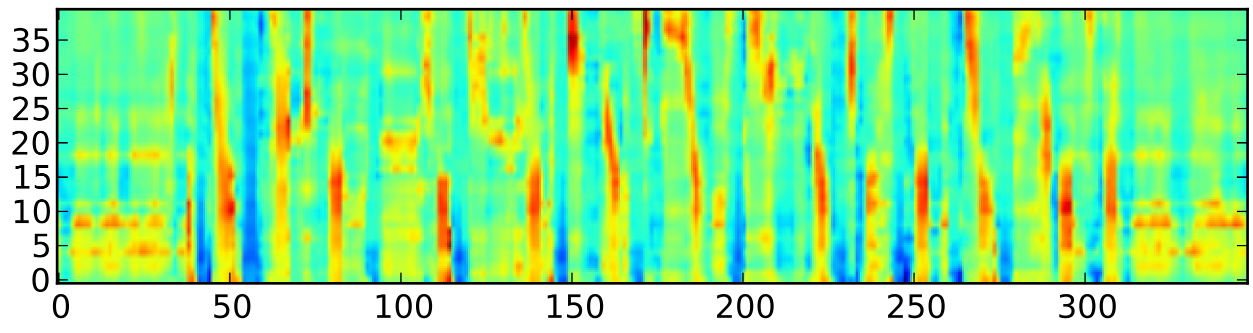
We evaluated the proposed methods through two domain adaptation tasks, namely, noise-robust speech recognition and speaking style adaptation.

3.3.1 Noise-robust speech recognition

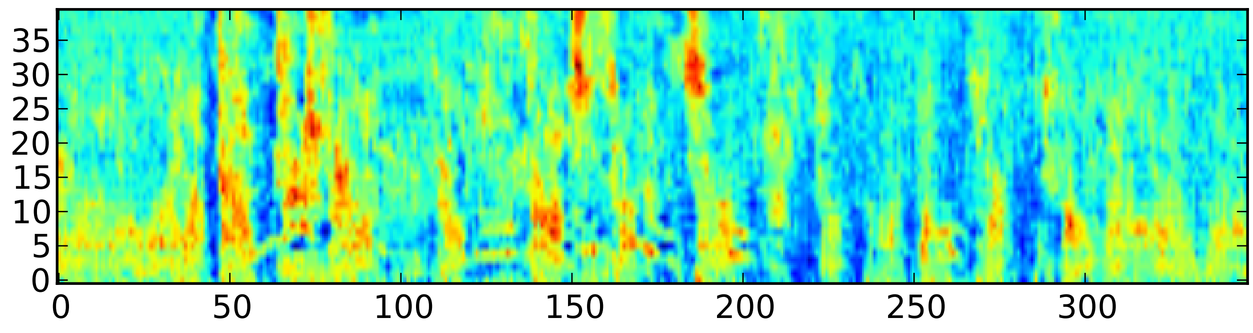
First, we evaluate the proposed method on a noisy speech recognition task, specifically the 1-channel track of the fourth CHiME Challenge [47], where the source domain is clean speech and target domain is noisy speech. CHiME3 is a data set for evaluating robust speech recognition systems. The "source" clean training set consists of 7,138 utterances from WSJ0 corpus. The "target" noisy training set consists of 1,600 real noisy utterances and 7,138 simulated noisy utterances generated by artificially mixing the clean training set with noise backgrounds. There are four different types of noisy environments, namely, bus, street, cafe, and pedestrian area [47]. A 440-dimensional feature vector consisting of 11 frames of 40-channel lmfb outputs is used as input to domain translation networks, as described in Section 3.1. The acoustic feature vectors in each set are normalized to have a zero mean and unit variance, and shuffled at frame level. For simplifying our training procedure, we used the same amount of shuffled data for both domains. We trained a CNN-HMM acoustic model [58] using the clean training set described above. It has two stride-1 convolutional layers, three fully-connected layers with 2k rectified linear units (ReLUs) [59] and a softmax output layer with 2k nodes. Each convolutional layer is followed by a stride-2 max pooling layer. The first convolutional layer has 180 filters of size 5x11, and the second one has 180 filters of size 5x1. The same 440-dimensional lmfb-based feature vector is used as input to the acoustic model as used for the domain translation networks, and we can directly input the translated features to the acoustic model. For decoding, we used the Kaldi



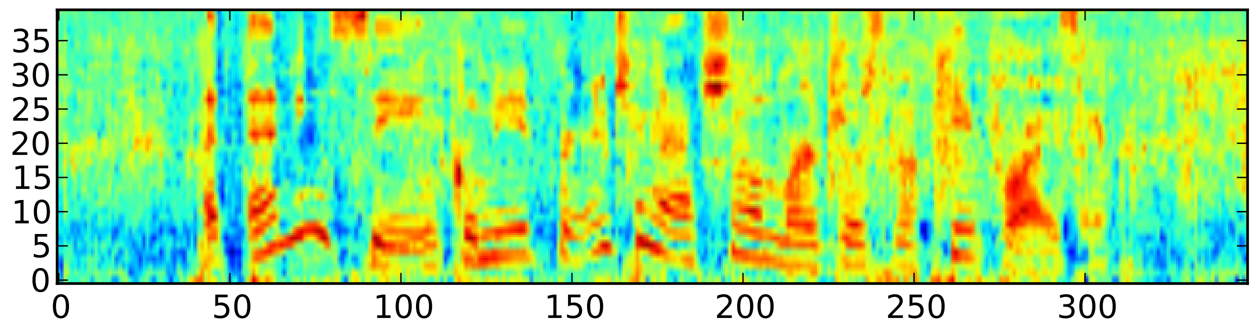
(a) original noisy data



(b) translated "fake" clean data with standard GAN

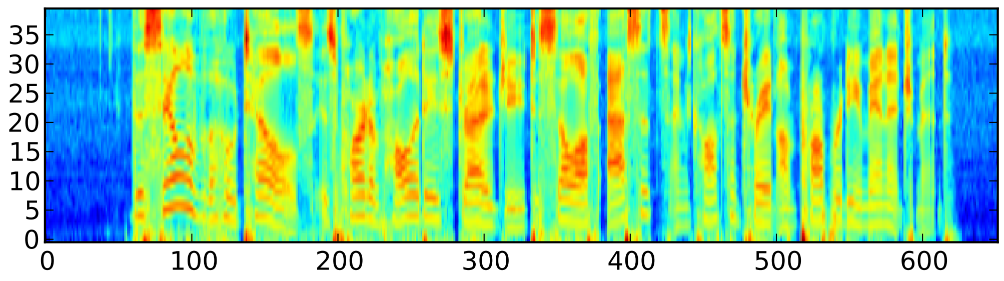


(c) translated "fake" clean data with standard GAN with cycle loss

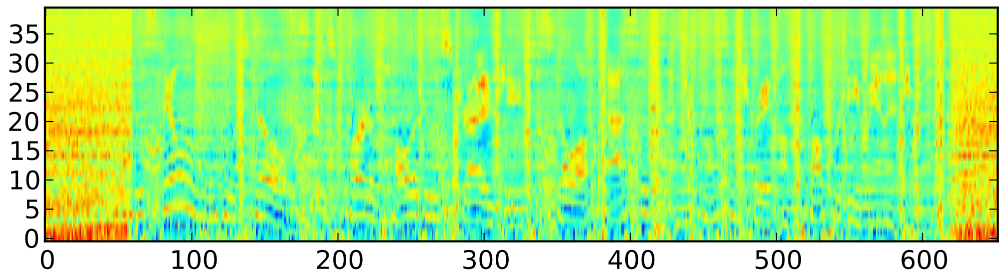


(d) translated "fake" clean data with proposed method

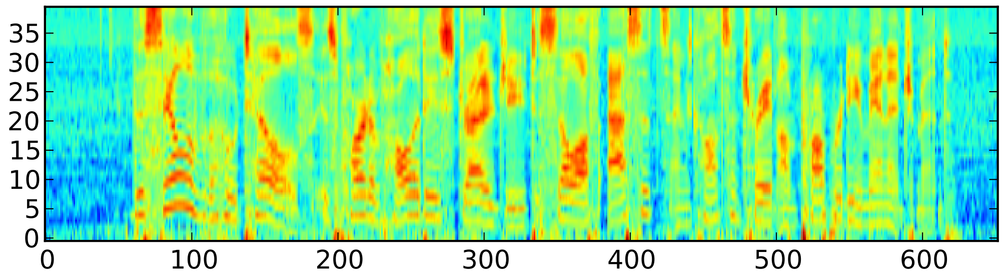
Figure 3.5: An example of noisy utterance and the translated "fake" clean data with various methods



(a) original clean data $\boldsymbol{s}$



(b) $F_{S\to T}(\boldsymbol{s})$



(c) $G_{S\to T}(\boldsymbol{s}) = F_{S\to T}(\boldsymbol{s}) + \boldsymbol{s}$

Figure 3.6: An example of clean utterance and the translated "fake" noisy data with the proposed method

WFST decoder [60]. The language model is the standard WSJ 5k trigram LM. We used the real noisy evaluation set ("et05_real_noisy") consisting of 1,320 utterances for evaluating the methods.

We present some examples of comparative domain translation results with the proposed approaches. Fig. 3.5 depicts an lmfb spectrogram of noisy speech and its domain transformation results using three different methods[2]. Fig. 3.5 (b) is the "fake" clean speech translated from the original noisy speech (a), using $G_{T\to S}$ trained only with the Wasserstein adversarial loss. Note that when the cycle consistency loss is not used, $G_{S\to T}$ and $G_{T\to S}$ are trained independently,

[2]Although a domain translation output vector consists of 11 frames of 40-channel lmfb features, we show here only the sequence of the center frames.

Table 3.1: Performance of proposed methods (WER(%))

acoustic model	feature	cycle loss	λ and μ	WER	ID
no adapt.	no adapt.	-	-	41.08	(1)
no adapt.	adapt. with $G_{T\to S}$	no	1, 1	55.45	(2)
		yes	1, 1	37.34	(3)
		yes	trained	36.56	(4)
adapt. with $G_{S\to T}$	no adapt.	yes	1, 1	35.98	(5)
		yes	trained	34.31	(6)

because they cannot affect each other. Obviously, (b) is totally different from the original utterance (a), and it hardly looks like human speech since it does not have formant trajectories. Compared with (b), the translated utterance with a GAN trained using the cycle consistency loss (c) has much more detailed structures as expected. It looks "cleaner" than (a) because it has more blue regions, and it is much more like human speech than (b). However, the apparent phonetic structure in the original data seems to be totally lost in (c), which is fatal for ASR. (d) is the translated data with a GAN which has our proposed network architecture and was trained using the cycle consistency loss. Noise is effectively suppressed and speech is enhanced here, and more importantly, a consonant-vowel phonetic structure is clearly seen in the translated spectrogram.

In Fig. 3.6, we show a "fake" noisy data translated from a clean utterance using $G_{S\to T}$. We separately present the outputs of two subnetworks (cf. Fig. 3.2) in order to show the mixing process. (a) is the output of the identity mapping component, which is of course identical to the input s. (b) is the output of the generative network component $F_{S\to T}(\boldsymbol{s})$, and (c) is the resulting noisy version of the input utterance generated by simply summing (a) and (b) together. In this example, we chose the network trained with fixed weights, $\lambda = \mu = 1$, for the purpose of demonstration. We can see that our method can make up a quite realistic noisy utterance which has similar characteristics to a real noisy data such as (a) in Fig. 3.5.

We show the speech recognition results by our proposed methods in Table 3.1. First, we evaluated the performance of speech enhancement using the noisy to clean mapping $G_{T\to S}$. Speech recognition was performed using the acoustic features of the real noisy test set transformed using

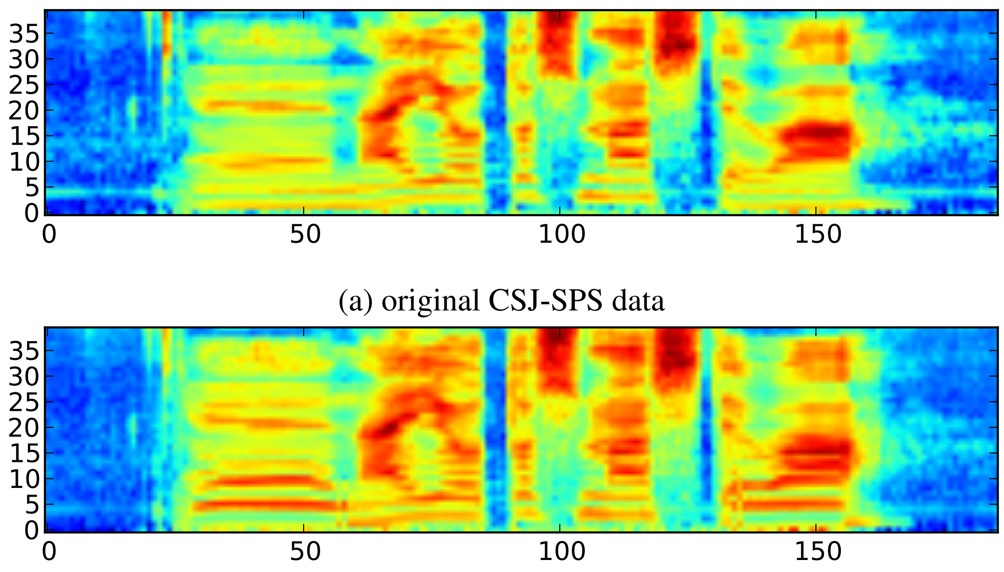
$G_{T\to S}$ and the baseline acoustic model trained using the clean data. By comparing the results for the original noisy data (row (1)) and the enhanced data (row (3)), we see that the proposed method effectively enhanced the acoustic features and yielded an improvement of 3.7 points in WER. Moreover, by introducing the submapping-wise scaling factors $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$, we had a further improvement of 0.78 points (row (4)). We also show the result for a GAN with our proposed architecture trained without the cyle consistency loss in row (2), from which we understand that the constraint that the mappings are cycle-consistent is essential for the desired quality in the translated features. Next, we evaluated the ASR performances of the adapted acoustic models. These models were trained using the "fake" noisy data translated with $G_{S\to T}$. From the results in row (5) and (6), we understand that the model adaptation approach is more effective than the feature enhancement approach (row (3) and (4)), and training $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$ is also beneficial for adapting acoustic models. The best model achieved WER of 34.35 % which is a 16 % relative improvement from the baseline. We see this outcome quite promising as a result of our first trial of unsupervised domain mapping, even though it is not yet comparable to the WER of 21.15 % which is achieved by the traditional training approach in which we make direct use of labeled simulated noisy data for acoustic model training.

3.3.2 Speaking style adaptation

We also applied the proposed method to speaking style adaptation of acoustic models to improve the ASR performance for test data with a different speaking style from the training data.

We used three corpora with different speaking styles, namely, JNAS (Japanese Newspaper Article Sentences), and APS (Academic Public Speaking) and SPS (Simulated Public Speaking) subcorpora from the CSJ (Corpus of Spontaneous Japanese). JNAS is a read speech corpus, and utterances in CSJ-APS and CSJ-SPS have spontaneous speaking styles. CSJ-APS consists of live recordings of academic presentations in public, and speeches in CSJ-SPS were presented in front of a small audience and in a relatively relaxed atmosphere. Speaking style transformation experiments were conducted using JNAS and CSJ-APS as source domains and CSJ-SPS as a target domain. We used the same amount of data (20 hours) for each corpus in training the GANs.

Fig. 3.7 depicts a lmfb spectrogram of a CSJ-SPS utterance transformed to JNAS style.



(a) original CSJ-SPS data

(b) translated "fake" JNAS data

Figure 3.7: An example of CSJ-SPS (simulated public speech) utterance and the translated "fake" JNAS (read speech) data with the proposed method

Table 3.2: Performance of proposed methods applied to speaking style adaptation (WER(%))

source	target	feature	WER
JNAS	CSJ-SPS	no adapt.	26.47
		adapt. with $G_{T\to S}$	25.93
CSJ-APS	CSJ-SPS	no adapt.	17.15
		adapt. with $G_{T\to S}$	16.60

While a number of differences between the original and transformed data can be observed, the most evident one is the enhanced formant structures in the region from the 20th to around the 50th frame, which corresponds to a filler word consisting of one long vowel "e:". It is a natural consequence considering that a more articulately pronounced vowel is a characteristic of read speech.

We present speech recognition experiment results using the adapted acoustic features in Table 3.2. The acoustic models were trained using source domain data. The WERs were improved by 0.5 points by adapting acoustic features to the source domains. The most remarkable point is that these improvements were obtained using only acoustic signals of the target domain without any supervision labels. From these results, we understand that our method can be applied to speaking style adaptation as well as noise-robust acoustic model training. Note that we did not conduct acoustic model adaptation with $G_{S \to T}$ in the speaking style translation experiments due to the limited time.

3.4 Conclusion

We proposed a novel approach to noise-robust acoustic training with GANs which are trained with a cycle consistent loss and have a specially designed architecture for retaining discriminative information in translated data. We demonstrated the effectiveness of the proposed method in noisy speech recognition and speaking style adaptation.

This is our initial attempt to apply deep generative networks for speech recognition. We are interested in extensions of the proposed methods such as introduction of recurrent structures for incorporating longer context information. Another promising direction is to use class information in training GANs [61] to enhance discriminability in translated data.

Chapter 4

Leveraging sequence-to-sequence speech synthesis for enhancing acoustic-to-word speech recognition

4.1 Introduction

We have seen a rapid development of alternative sequence-to-sequence (seq2seq) approaches to speech recognition based on connectionist temporal classification (CTC) [28, 62, 63, 64], attention-based encoder-decoder models [30, 37, 31, 65] and RNN-transducers [29, 66]. Their remarkable advantage is that they get rid of dependency on frame-level probabilistic state transition models such as HMMs. An extreme example of the seq2seq approach is acoustic-to-word (A2W) models [67, 68, 69] which directly map acoustic signals into word sequences. They do not require any external decoders, leading to an extremely simplified architecture of ASR systems and very low latency. Outputting words rather than phones or characters is also an advantage since it requires no post-processing to utilize the ASR output in the subsequent natural language processing such as dialogue, translation and information query.

We have shown in [69] that attention-based models in which the label output probability is explicitly conditioned on the past output are significantly better than CTC-based models for word-level seq2seq speech recognition. We have also demonstrated in [70] that an attention-based A2W model achieved a WER reduction of 25.3% relative from a state-of-the-art hybrid

DNN-HMM system with a decoding speed faster by a factor of 50. In this chapter, we further seek to improve the attention-based A2W speech recognition system.

Although A2W models are attractive for multiple reasons we pointed out above, they have some drawbacks compared to conventional systems based on phones or characters. The most important problem is that they cannot predict posterior probabilities for unknown words which did not appear in the training data and have no mechanism to add new words to its vocabulary. This is a serious problem, since an ASR system may encounter a number of words specific to the domain which are out of vocabulary (OOV) of the system, when it is deployed in a particular task domain.

In addition to this OOV word problem, there is another issue that A2W models have no way to utilize text data directly to improve its language modeling capability. It is because A2W models are trained in a seq2seq manner from pairs of speech and word labels. In other words, they are constrained to learn a probability distribution over word sequences only from a limited amount of labeled speech, although it could be estimated more reliably from a large collection of texts covering many linguistic phenomena. It also implies that we cannot perform domain adaption of A2W models using relevant texts in a target domain.

Recently, sequence-to-sequence neural speech synthesis models trainable from corpora have been developed and shown to achieve naturalness comparable to recorded human speech [71, 72, 73]. In this chapter, we propose a novel approach to address the problems inherent to A2W models exploiting the current seq2seq speech synthesis technologies. In this approach, we perform training data augmentation by generating speech features from a set of target domain texts using a seq2seq speech synthesizer. These artificial speech features together with real speech features from conventional speech corpora are used to train an attention-based A2W model. We only need to prepare relevant texts to the application domain of the speech recognition system, which are much more easily accessible than labeled speech data. We can expand the vocabulary and enhance the language modeling capability of the A2W model using new words and word contexts included in the augmented data. Moreover, our method makes it possible to integrate an A2W model with an RNN-based external language model in a natural way by ensuring that they have the same vocabulary. We also explore an encoder freezing learning technique to prevent the undesirable effects from the uniformness of synthesized speech.

The experimental evaluations show that the proposed approach implemented with a speech synthesizer trained using speech data from a single female speaker significantly improved the speech recognition performance compared to the baseline models trained using only the real speech data.

4.2 Sequence-to-sequence speech synthesis

Seq2seq speech synthesis is a technology for generating speech directly from text which does not require complex multistage pipelines unlike traditional text-to-speech (TTS) systems. Although a number of distinct architectures have been proposed [74, 71, 72, 73], these systems are commonly composed of an attention-based feature prediction network which maps character embedding to mel-scale spectrograms or vocoder parameters, followed by a vocoder which synthesize time-domain waveforms from these predicted features. In this research, we use Tacotron 2 [72] which has a relatively simple architecture similar to our A2W model and was shown to achieve striking naturalness. Here, we describe in depth the feature prediction network of Tactron 2 along with the network configurations we used in our experiments. Note that we do not need a vocoder, since what we need is mel-scale spectrograms for training a recognizer rather than waveforms.

The feature prediction network consists of the character encoder and the attention-based decoder subnetworks. The former summarizes the input character sequence and outputs a sequential representation of the same length as the input sequence. The latter predicts a sequence of Mel-spectrograms in an autoregressive way conditioned on the encoder outputs.

In the encoder subnetwork, each input character is first mapped to a 512-dimensional continuous vector. This mapping is performed via a learnable character embedding layer. These character embeddings are fed to a stack of 3 convolutional layers. Each layer convolves 512 filters of size $(5, 1)$ to its input, followed by batch normalization [75] and ReLU activations. The output of the final convolutional layer is input to a one-layer bidirectional LSTM with 256 memory cells in each direction to generate the encoded features.

The decoder subnetwork predicts five consecutive frames of Mel-spectrograms at each decoding step based on the encoder outputs and the final frame of the predicted features at the previous step, as follows. The encoder outputs are summarized using the location-sensitive attention mechanism [37]. The attention weight at each decoding step is calculated using the

128-dimensional projected vectors of the decoder LSTM state, the encoder output sequence and the location features. The location features are calculated by convolving 32 one-dimensional convolution filters with length 31 to the cumulative vector of the attention weights in all past decoding steps. The sum of these vectors is normalized using a formula equivalent to (4) in the last section after applying tanh activation to generate the attention weight of the current decoding step. Meanwhile, the last one frame of the prediction in the last time step is passed through a pre-net consisting of two fully-connected layers with 256 ReLU units. This pre-net output and the attention vector are concateneted to be provided to a 2-layer unidirectional LSTM with 1024 memory cells. The LSTM outputs together with the attention context vector are passed through a linear projection layer to predict the 5 frames of the target Mel-spectrograms.

After finishing all decoding steps, the predicted Mel-spectrogram sequence are processed by a 5-layer convolutional post-net which predicts a residual to add to the original prediction. Each convolutional layer has 512 filters of size $(5, 1)$. Batch normalization and tanh activations are applied on all but the final layer. The entire network is trained using the L1 distance between the predictions and the target spectrograms as the loss function[1].

The decoder simultaneously predicts if the output sequence has completed or not at each time step. It is judged based on the decoder LSTM output and the attention context. More precisely, the concatenation of these vectors are projected down to a scalar and applied with sigmoid activation to calculate the probability that the decoding is complete. In the runtime of speech synthesis, we stop generating acoustic features if this probability exceeds 0.5.

Seq2seq speech synthesis techniques are commonly shown to achieve very high mean opinion scores (MOS) using unified simple architectures which are easy to implement and optimize. They also have an advantage that they generate a sequence of acoustic features we can use directly as the input to speech recognition systems. These are the reasons why we adopt seq2seq speech synthesis in our proposed approach instead of conventional unit selection-based concatenative methods or HMM-based statistical parametric methods.

[1]The original Tacotron 2 used the mean squared error (MSE), which gave slightly worse results than the L1 loss in our preliminary experiments.

4.3 Proposed method

Before introducing our proposed methods, we recap the problems with A2W models. First, they cannot predict posterior probabilities of words which did not appear in the training data. Therefore, they are never able to recognize a sentence with these OOV words correctly. Second, since the entire network of an A2W model is trained in a seq2seq manner from pairs of speech and word sequences, there is no direct way to enhance language modeling capability of the model even if a large collection of texts is available.

A simple way to address the OOV word problem is to combine A2W models with character-based models [76, 69]. In this approach, speech is decoded using both of an A2W model and an acoustic-to-character (A2C) model. When an OOV word is detected with the A2W model, the character sequence from the A2C model appearing in the speech segment corresponding to the OOV word is output instead of the OOV symbol. These methods assume that OOV words are detected with a high recall rate. A more recent approach called the modular training of A2W models aims to resolve both of the two problems [77]. An A2W model is factorized into two modules, an acoustic-to-phoneme (A2P) model and a phoneme-to-word (P2W) model. This is similar to the traditional hybrid systems which combine acoustic, transition, pronunciation and language models in a modular way. The P2W model maps phone sequences to word sequences and can be trained from text data without speech.

In this research, we propose a more direct approach for enhancing A2W models inspired by the recent progress in seq2seq speech synthesis. Our method makes it possible to train A2W models from arbitrary sentences leveraging a state-of-the-art speech synthesis technique. It does not require any modification to the simple architecture of A2W models or additional lower-level models retaining the full strength of vanilla A2W models.

4.3.1 Training data augmentation

We exploit seq2seq speech synthesis reviewed in the previous section for augmenting training data for attention-based A2W models using relevant texts to a target domain. Addition of the text with right words and word contexts unseen to the baseline model, trained only with an available real speech corpus, will contribute to expand the vocabulary and improve the language modeling capability of the A2W model.

In our first attempt of the proposed approach in this research, we work with the Japanese language. Since the number of distinct characters in Japanese are a few thousands, unlike languages such as English where the number is a few tens, it is obvious that we will have too many number of parameters in the model as well as a serious sparse data problem if characters are used as input units. Therefore, we opted to choose phones as the input unit rather than characters in the original Tacotron2.

We use a 40-dimensional vector consisting of 40-channel log Mel-filterbank (lmfb) outputs as the target of the synthesizer network, which is the same acoustic feature we use for our speech recognition systems. This is because we want to use an output sequence from the synthesizer directly as input to the A2W model in order to avoid an artifact caused by performing additional feature conversion. The frame window length and frame shift are set to be 10 ms and 25 ms following the standard setting in speech recognition.

The procedure of the data augmentation is as follows. We collect texts from a target domain where we want to perform speech recognition. Since words are not separated by spaces in Japanese, each sentence in the collected data is first processed by a Japanese morphological analyzer to separate it into words and simultaneously obtain the pronunciation of each word. The sequence of phones and special symbols representing word boundaries are fed into the seq2seq speech synthesizer to generate a mel-spectrogram for the sentence. The set of synthesized log mel-spectral features and corresponding word sequences are added to the conventional training data coming from the real speech corpora to train the attention-based A2W speech recognition model.

4.3.2 Encoder freezing learning of attention model

One concern with the data augmentation using a speech synthesizer is the possibility that artificial speech is much less acoustically diverse than real speech. This is more likely when the synthesizer is trained using a typical speech synthesis corpus consisting of voices from a single speaker. We thought that the augmented artificial data can be harmful for training the acoustic encoder subnetwork of A2W models, while they should be essential for enhancing the decoder.

Therefore, we adopted the *encoder freezing learning* of attention-based models which we investigated for domain transfer learning of A2W models [78]. In this framework, the parameters of the acoustic encoder are copied from a model pretrained on real speech, and they are fixed during the training using the augmented data set consisting of the artificial and real data. In other words, the encoder subnetwork for summarizing the acoustic information is trained only on real speech data, while the decoder layer which is responsible for predicting word transition probabilities is tuned using the full set of the augmented data. We show the procedure of the encoder freezing learning in Fig. 6.4. The decoder is designed to have an softmax output layer whose size is the same as the expanded vocabulary and its parameters are initialized with random values. This framework aims to prevent the undesirable effects from the uniformness of synthesized speech, while taking advantage of the diversity coming from a large text corpus.

4.3.3 Language model integration

It was shown to be very effective to incorporate external language models in speech recognition using conventional seq2seq models based on characters [79, 80, 81]. This is because language models can be trained on a large set of texts covering much richer linguistic information than a limited amount of labeled speech data used for training the seq2seq models.

On the other hand, integration of A2W models and external word-level language models has not been investigated well, mainly because the vocabularies of the A2W model and the language model trained on larger data are inevitably different. It is far from trivial to combine them to calculate scores for words in each decoding step. However, based on our data augmentation method, we can easily achieve a shallow fusion of the A2W model and the language model, because we can train both of them on the same word sequences and ensure that they have the same vocabulary. For decoding with the external language model, a small modification is required to line 12 in Algorithm 1 as:

$$\begin{aligned} score(\boldsymbol{s}^{+}) = score(\boldsymbol{s}) &+ \log(p(y|\boldsymbol{s}, \boldsymbol{X})) \\ &+\mu \log(p_{LM}(y|\boldsymbol{s})), \end{aligned} \tag{4.1}$$

where $p_{LM}(y|\boldsymbol{s})$ is the posterior probability of word y given word history $\boldsymbol{s}$, which is calculated with a LSTM-based neural language model. It is trained using all texts from the source and target domain. μ is the weight for the language model score. The A2W model and the external language model use context in different ways and are expected to complementarily contribute

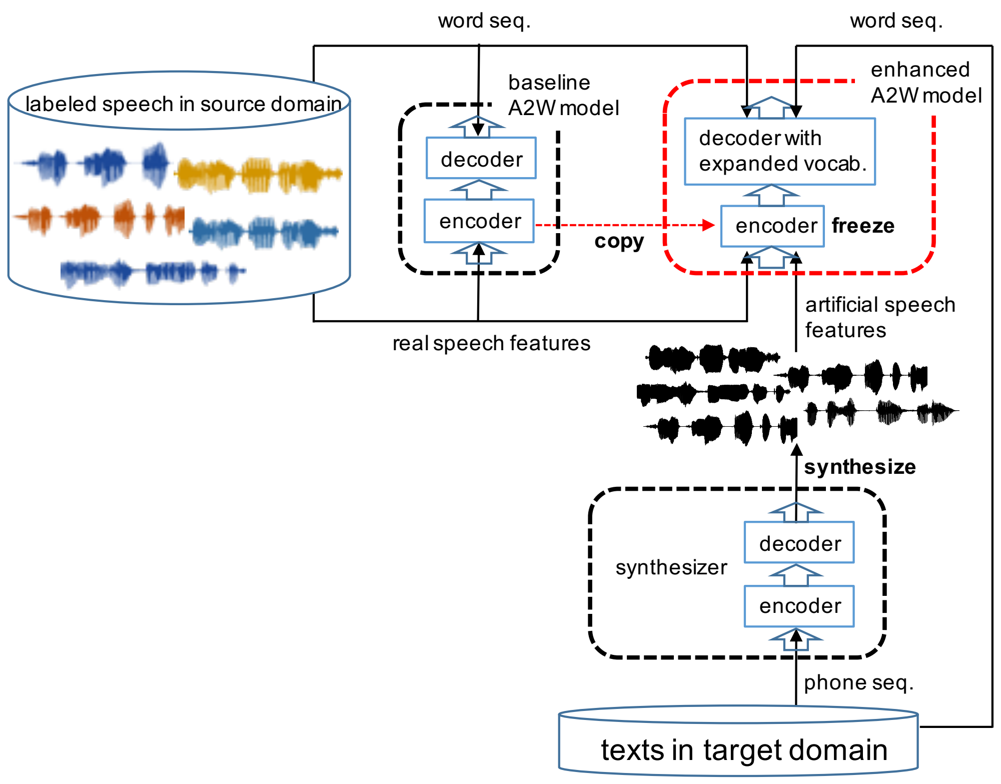


Figure 4.1: Encoder freezing learning for enhancing A2W model using real and artificial training data

to improve speech recognition performance[2].

4.4 Experimental evaluations

4.4.1 Data sets

We evaluated our methods through speech recognition tasks using two standard Japanese corpora: the Corpus of Spontaneous Japanese (CSJ) [82] and Japanese Newspaper Article Sentences (JNAS). CSJ includes two distinct subcorpora, namely, CSJ-APS and CSJ-SPS. CSJ-APS consists of academic presentation speeches on several topics such as science, engineering, humanities and social science. CSJ-SPS consists of simulated presentation speeches on three general themes. These subsets have their own official test sets, namely, CSJ-TESTSET1 and CSJ-TESTSET3. While CSJ comprises spontaneous utterances, JNAS consists of newspaper articles read aloud.

[2]For example, A2W models tend to give similar probabilities to words with similar pronunciations, while language models make predictions independent of word pronunciations.

4.4.2 A2W model

A 40-dimensional vector consisting of 40-channel log Mel-scale filterbank (lmfb) outputs is used as acoustic features for attention-based A2W models. Non-overlapping frame stacking [63] was applied to these features in which we stack and skip three frames to make a new super frame. The acoustic encoders in our attention models consist of 5-layers of bidirectional LSTMs with 320 cells. Dropout [83] was used for training each LSTM layer with a dropout rate of 0.2. The decoder consists of a 1-layer LSTM with 320 cells and a softmax output layer with the nodes for vocabulary words, $\langle\text{sos}\rangle$, $\langle\text{eos}\rangle$ and $\langle\text{OOV}\rangle$ special token for words which appeared less than 3 times in the training sets. The vocabularies of the A2W models consist of words which appeared more than two times in the training sets. We used Adam [57] for optimizing network parameters. We also used gradient clipping with a threshold of 5.0. All network parameters were initialized with random values drawn from a uniform distribution with range (-0.1, 0.1). We also used scheduled sampling [84] and label smoothing [85] to improve the optimization. In the experiments for integrating A2W models and RNN-based external language models, we used neural language models with 3 layers of unidirectional LSTMs with 256 memory cells. Each word is mapped to a 512-dimensional continuous vector before fed to LSTMs. We used PyTorch [86] to implement the A2W models and the LSTM-based language models. The beam width B was set to be 4 in all speech recognition experiments.

4.4.3 Seq2seq speech synthesizer

As we described in section 4.1, The seq2seq speech synthesizer is trained to output 40-channel log mel-filterbank features. For word segmentation and pronunciation annotation of texts, we used Mecab[3], a CRF-based Japanese morphological analyzer, combined with the "Unidic" word dictionary[4]. As training data, we adopted JSUT (Japanese speech corpus of Saruwatari-lab., University of Tokyo) corpus [87]. It is a recording of 7,607 prompt texts read aloud by a female speaker with total duration of ten hours. The prompt sentences were tokenized using Mecab and the corresponding phone sequences were obtained. These automatically generated phone labels are used without human checking and, therefore, are expected to include a certain amount of

[3]http://taku910.github.io/mecab/

[4]http://unidic.ninjal.ac.jp/

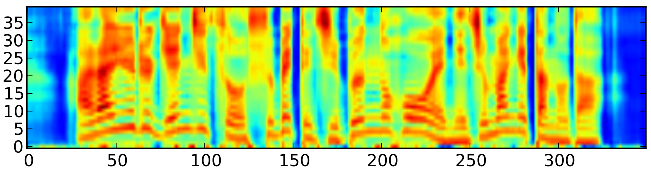


Figure 4.2: An example of synthesized lmfb features from a Japanese sentence *"arayuru geNjitsu o subete jibuN no ho: e nejimage ta no da"*

erroneous ones. We used 33 phone classes including special tokens for pause, word boundary and the end of a sentence. We also used PyTorch [86] to implement the Tacotron2-based feature prediction network using phones as input units. We used dropout with a dropout rate of 0.5 in all convolutional layers following the recipe in [72]. The LSTM layers in the decoder subnetwork are regularized using zoneout [88] with a probability of 0.1. As in the training of A2W models, we used the Adam optimizer. Fig. 4.2 depicts an example of speech synthesized from a sentence in the training set of JNAS [5].

4.4.4 Adaptation between two spontaneous speech domains

We first examined how proposed approach works effectively in adapting a seq2seq ASR model trained with spoken presentations in general everyday life topic domain to the academic presentation domain on audio, speech and language processing, using SPS ("Simulated" Presentation Speech) and APS (Academic Presentation Speech) subcorpora of the CSJ corpus.

The baseline model was trained using SPS training set consisting of 281 hours of spontaneous speech. Its vocabulary consisting of all the distinct words occurring more than twice in the training set comprises 24,826 words. For the adaptation for Academic presentation domain, we synthesized 282,235 utterances from the transcript of the APS training set using the seq2seq speech synthesis model of single female voice described in the last subsection. The ASR model for the target domain was trained by adding this 255 hours of synthetic speech to the baseline 281-hour SPS training set. By the addition of the synthetic APS set, the vocabulary of the model was expanded by around 10k words and reached 34,331 words. The training was performed

[5]We cannot assess the naturalness of the synthesized features directly, because it is difficult to reconstruct waveforms from 40-dimensional lmfb features. However, we found that the synthesized utterances were recognized almost perfectly using an A2W model trained on real speech.

Table 4.1: ASR performance for two CSJ test sets (WER(%))

	TESTSET1 (APS)	TESTSET3 (SPS)
SPS (real) (baseline)	24.92	11.43
+ language model (SPS)	25.37	11.27
SPS (real) + APS (synthesized)	**19.40**	**10.42**
+ language model (APS + SPS)	**18.74**	**10.22**
SPS (real) + APS (synthesized) with enc. freezing	**19.09**	**11.29**
+ language model (APS + SPS)	**18.38**	**11.07**

with and without encoder freezing described in Section 4.2.

The second column in Table 4.1 ("TESTSET1 (APS)") shows the result for the target domain test set[6]. TESTSET1 is composed of ten academic presentation speeches by ten male speakers. The baseline model trained only with the SPS training data gave a relatively high WER of 24.9% and integration of the language model trained on the SPS training set transcript rather degraded the accuracy slightly. On the other hand, with the enhanced model trained by the proposed approach using synthetic speech, the WER for the target domain test set was reduced by 5.5 points to 19.4%. The encoder freezing gave a further reduction of 0.31 points. The LSTM language model trained with the transcripts of both SPS and APS training data consistently yielded significant improvements. The best model gave an WER of 18.38%, which corresponds to a 26.2% relative improvement over the baseline. We observed that a number of words unknown to the baseline model were correctly recognized with the enhanced models with a enlarged vocabulary. These words include, for example, "efuzero" (F0) and "oNcho" (tone) that often appears in the context of audio, speech, and language processing. It is noteworthy that we had these substantial improvements with the all male-speaker test set by augmenting the training data with synthesized speech of a single female voice. It may be because the synthetic speech by the seq2seq synthesizer has a high enough naturalness to contribute to the training of the decoder part of the A2W model as well as it does not hurt the training of the encoder part of the model which has a inherent discriminative nature resistant to the quantitative dominance of a single speaker.

[6]The OOV rates of TESTSET1 for the baseline and enhanced models are 4.48% and 0.96%, respectively.

We also looked at the influence of the domain adaptation to the original source domain (SPS) test set and show the results on the third column of Table 4.1. The test set of CSJ-SPS, TESTSET3, is composed of ten simulated presentation speeches by five male and five female speakers. Although the WER obtained with the baseline model was already low, the data augmentation using artificial data gave a further significant improvement. We understand that this gain is due to the improved capability for estimating word transition probabilities learned from enhanced training data based on a considerable amount of text. However, a different tendency was observed in the encoder freezing learning compared to the cross-domain results. For the SPS test set, encoder freezing training gave an improvement from the baseline, but was not as good as training of the whole model. Again, the language model integration was effective for both of the enhanced models.

4.4.5 Adaptation to newspaper domain

We also attempted a more challenging domain adaptation from spontaneous presentation to read speech of newspaper articles, where the difficulty comes from the large differences in speaking style and information content. This time, the baseline model was trained using the whole real speech training data of the CSJ corpus comprising both of APS and SPS. Its vocabulary consists of 32,573 words. For the target domain ASR model, we synthesized a set of utterances from 49,576 news paper prompt texts of the JNAS training data and added them to CSJ training set to train the A2W model. By the addition of synthetic JNAS data, its vocabulary turned out to be 35,795 words. In order to manage the consistency of word boundary definitions in using two distinct corpora, which is not obvious as a matter of fact with the Japanese language, we tokenized and generated pronunciations of the both corpora using Mecab. The label error rate of the automatically generated phone sequences calculated using the manual transcriptions as reference was 6.29%. The results of the speech recognition experiment using the test set of JNAS are summarized in Table 2. The test set is composed of 200 sentences spoken by 22 male and 22 female speakers.

The enhanced model with the proposed method gave a much lower WER than the baseline trained on all utterances in CSJ. We also observed an interesting fact that the encoder freezing learning was quite effective in this experiment which gave an additional improvement of 2.2

Table 4.2: ASR performance for JNAS test set

	training data amount (hours)
CSJ (real) (baseline)	528.8
JNAS (real) (oracle)	85.5
CSJ (real) + JNAS (real) (oracle)	614.2
CSJ (real) + JNAS (synthesized)	596.5
CSJ (real) + JNAS (synthesized) with enc. freezing	596.5
+ language model (CSJ + JNAS)	-
CSJ (real) + JNAS (synthesized) + Mainichi (synthesized) with enc. freezing	1502.10
+ language model (CSJ + JNAS + Mainichi)	-

points. One possible reason for this is that the speaking style of the test set speech is more different from the added synthetic speech than what we expect about two sets that are both coming from read speech. Integrating the external language model also yielded a WER reduction, resulting in an improvement of 50.1% relative over the baseline.

We also tried to further increase the amount of training data using an external language resource. We randomly picked 500k sentences from articles of Mainichi Shimbun, one of the major newspapers in Japan, published in 2000 and 2001. The artificial speech features synthesized from these newspaper sentences are used for training a new A2W model together with CSJ and the artificial data generated from sentences of JNAS. The vocabulary size of the new model is 41,890. By performing this additional data augmentation, the WER was further reduced by 1.23 points as shown in the seventh row in Table 2. The language model integration was also effective. From these results, we confirm that relevant texts from outside of the target corpus can be utilized to improve the performance of the A2W model as well.

For comparison, we also built two oracle models. One is the model trained using the real utterances in JNAS, which gave a poor speech recognition performance due to the small amount of training data. The other is the A2W model trained on real data from both of CSJ and JNAS. This model yielded a very low WER. It is encouraging to note that the enhanced model using artificial augmented data with our proposed method gave a gain corresponding to as large as

70% of the gain coming from adding the true speech data of the domain in this oracle model.

4.5 Conclusion

We showed that we can significantly enhance the speech recognition performance of A2W models using only text data via phone-based seq2seq speech synthesis. The proposed method makes it possible to train an A2W model from arbitrary sentences and effectively expand the vocabulary and improve language modeling capability of the model. We demonstrated the effectiveness of the method thorough two cross-domain speech recognition experiments.

Previously a combination of seq2seq speech synthesis and recognition has been investigated in [89]. The novelty of our contribution is that we exploited seq2seq speech synthesis for enhancing the A2W model and demonstrated that we can significantly improve speech recognition performance using artificial training data, while the main issue in [89] was investigating a deep learning-based speech chain model.

Although our method extremely expands the vocabulary of the A2W model, it is still not totally free from the OOV word problem. It can be combined with existing methods for addressing problems of A2W models such as [76, 69, 77] to achieve a further improvement. While one of the important findings in this chapter is that we can significantly improve the performance of A2W models using a speech synthesizer trained on a typical speech synthesis corpus consisting of speech from a single speaker, it would achieve a further improvement if we can train a synthesizer on a large speech corpus containing many speakers and many speaking styles. Another possible direction is, for example, investigating joint training of the seq2seq synthesizer and the A2W model.

Chapter 5

Forward-backward attention decoder

5.1 Introduction

We have shown in [69] that attention-based models which explicitly incorporate label transition probabilities are significantly better than CTC-based models for word-level seq2seq speech recognition. In this chapter, we further seek to improve the attention-based speech recognition system. Specifically, we look at attending the encoded feature stream from right to left as well as left to right, and investigate how this backward attention can be utilized together with the usual forward attention to improve the performance of attention-based models. In the proposed approach, speech is decoded from right to left as well as from left to right utilizing forward and backward attention vectors, and the best sentence hypothesis is searched for based on combined probabilities provided by the decoders for two directions. We also propose a multitask learning in which the forward decoder is trained with a subtask of backward decoding sharing a single encoder and vice versa, aiming at a regularization effect of encoder optimization for both directions. We demonstrate that acoustic-to-word attention-based models enhanced with the proposed approach achieve much higher performances than a state-of-the-art hybrid DNN-HMM system with a decoding speed faster by a factor of 50 through speech recognition experiments using large-scale spontaneous Japanese corpora.

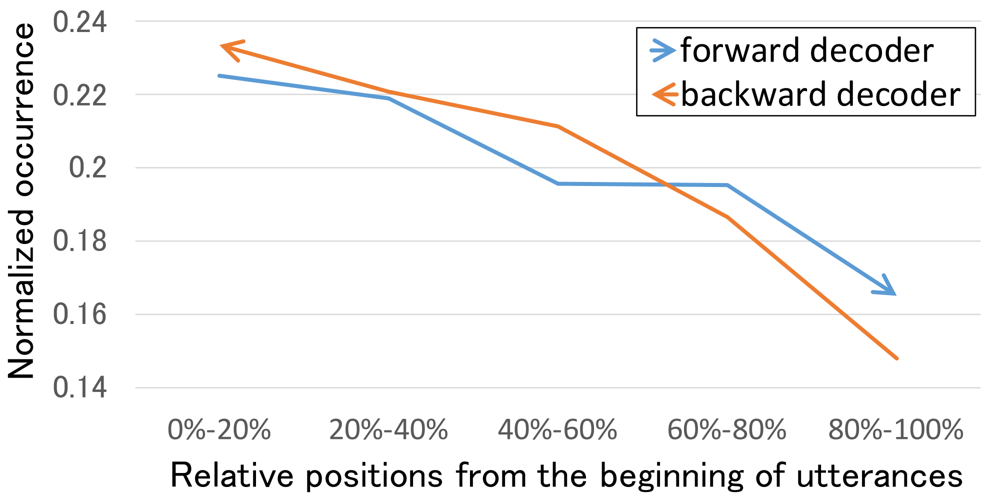


Figure 5.1: Normalized occurrence counts of substitution errors at different relative positions of utterances in CSJ-TESTSET1

5.2 Forward-backward attention

Here, we describe in depth the proposed method that integrate forward and backward attentions. This approach is motivated by the following two considerations.

First, attention-based encoder-decoder models are generally not good at decoding long utterances as mentioned in [37]. They tend to output less reliable labels in the latter part of utterances due to error accumulation. Figure 5.1 shows how many substitution errors word-level forward and backward decoders made at different relative positions of utterances in a Japanese speech recognition experiment [1]. In this figure, the forward decoder made more errors than the backward counterpart in the later part of utterances, while we observe an opposite tendency in the earlier part. This shows that unidirectional decoders tend to be more accurate at their early decoding steps, which leads to a strategy that concatenates reliable parts of sequences obtained from two directional decoders. These newly generated hypotheses can have fewer errors than the original ones, if this concatenation process is managed in an appropriate way.

Second, because speech has an asymmetric time structure, forward and backward decoders may have strengths in different ways which may contribute to better accuracy complementarily. Therefore, we expect a performance gain from simply combining their recognition results,

[1]This is the same experiment as shown in the first column of Table 5.1.

considering the consistent effectiveness of the system combination approaches [90][91] in many ASR tasks such as noisy speech recognition [92].

5.2.1 Decoding

Algorithm 2 ForwardBackwardBeamSearch($N, \boldsymbol{X}$)

```
1: $F^f \Leftarrow$ ForwardBeamSearch($N, \boldsymbol{X}$)
2: $F^b \Leftarrow$ BackwardBeamSearch($N, flip(\boldsymbol{X})$)
3: $F \Leftarrow \{\phi\}$
4: for $\boldsymbol{s}^f \in F^f$, $\boldsymbol{s}^b \in F^b$ do
5:   $j\_next = |\boldsymbol{s}^b|$
6:   for $i$ in 1,...,$|\boldsymbol{s}^f|$ do
7:     for $j$ in $j\_next$,...,1 do
8:       if $s^f_i = s^b_j$ and $time(s^f_i) \sim time(s^b_j)$ then
9:         $\boldsymbol{s}^* \Leftarrow concat(\boldsymbol{s}^f_{<=i}, flip(\boldsymbol{s}^b_{<j}))$
10:        $score(\boldsymbol{s}^*) = score^f(\boldsymbol{s}^f_{<i}) + score^b(\boldsymbol{s}^b_{<j}) + \log(\max(p^f(s^f_i|\boldsymbol{s}^f_{<i}, \boldsymbol{X}), p^b(s^b_j|\boldsymbol{s}^b_{<j}, flip(\boldsymbol{X})))$
11:        add $s^*$ to $F$
12:        $j\_next = j - 1$
13:        break
14:      end if
15:    end for
16:  end for
17: end for
18: return set of sentence candidates $F$
```

We propose a 3-pass decoding algorithm utilizing forward and backward attention decoders which do not undermine the low-latency advantage of word-level models. We summarize it in Algorithm 2. It was inspired by decoding methods for HMM acoustic models based on forward-backward search [93][94][95]. Pass 1 performs left-to-right decoding using the forward decoder. In Pass 2, speech is decoded from right to left using the backward decoder taking flipped acoustic features, which we denote as $flip(\boldsymbol{X})$ in Algorithm 2, as input. Pass 3 integrates the sentence hypotheses from Pass 1 and Pass 2 and generates new candidates by concatenating the partial sentences.

The key points for this algorithm are (1) how to find the partial sequences to be concatenated from hypotheses obtained with the two decoders, and (2) how to give reliable scores to the newly generated sentence candidates. As for the first issue, we first pick up two sentence hypotheses which share the same word appearing at approximately the same time frame. We then split each of these two sentence hypotheses at this word and concatenate the earlier half of the forward hypothesis and the later half of the backward hypothesis to generate a new sentence hypothesis.

More precisely, we judge that the i-th word s^f_i in a forward sentence candidate $\boldsymbol{s}^f$ and the j-th word s^b_j in a backward candidate $\boldsymbol{s}^b$ appear at an approximately same time frame (which we denote $time(s^f_i) \sim time(s^b_j)$), if the following condition is met:

$$time(s^b_{j+1}) < time(s^f_i) < time(s^b_{j-1}), \tag{5.1}$$

where we define the occurrence time of words using forward and backward attention vectors as:

$$time(s^f_i) \overset{\text{def}}{=} \arg\max(\boldsymbol{\alpha}^f_i), \tag{5.2}$$

$$time(s^b_j) \overset{\text{def}}{=} T - \arg\max(\boldsymbol{\alpha}^b_j), \tag{5.3}$$

where T is the frame length of the input speech. $\boldsymbol{\alpha}^f_i$ and $\boldsymbol{\alpha}^b_j$ are the forward attention vector at the i-th forward decoding step and the backward attention vector at the j-th backward decoding step, respectively. This is a natural choice because attention vectors have their maximum values around the center frame of the corresponding words in most cases.

The generation process of new sentence candidates is as follows. From any pair of forward and backward sentence hypotheses, we pick pairs of words whose occurrence time are potentially similar based on their positions in hypotheses. We examine if the condition given in equation (10) is true for the word pair and if these word labels are the same. If s^f_i and s^b_j satisfy the conditions, we generate a new sentence candidate $\boldsymbol{s}^*$ as:

$$\boldsymbol{s}^* = concat(\boldsymbol{s}^f_{<=i}, flip(\boldsymbol{s}^b_{<j})), \tag{5.4}$$

where $\boldsymbol{s}^f_{<=i}$ is a partial sequence beginning from $\langle\text{sos}\rangle$ and ending at the i-th word of the original forward hypothesis, and $\boldsymbol{s}^f_{<j}$ is a partial sequence beginning from $\langle\text{eos}\rangle$ and ending at the $(j-1)$-

th word of the original backward hypothesis[2]. Because these two partial sentences end at the same word appearing at an approximately same time frame, the score for this concatenated sequence $\boldsymbol{s}^*$ is defined as:

$$\begin{aligned} score(\boldsymbol{s}^*) = score^f(\boldsymbol{s}^f_{<i}) + score^b(\boldsymbol{s}^b_{<j}) + \\ \log(\max(p^f(s^f_i|\boldsymbol{s}^f_{<i}, \boldsymbol{X}), p^b(s^b_j|\boldsymbol{s}^b_{<j}, flip(\boldsymbol{X})))). \end{aligned} \tag{5.5}$$

Note that the double loop from line 6 to 16 runs in $\mathcal{O}(L)$ time where L is the length of the longest sentence candidate of F^f and F^b, since we only need to check the pairs of words from $\boldsymbol{s}^f_{>i}$ and $\boldsymbol{s}^b_{<j}$ after we confirm that s^f_i and s^b_j satisfy the condition on line 8. After all candidates are generated, we rescore them using a formula (9) and output the sentence with the largest score.

5.2.2 Multitask learning of decoders

We also propose to integrate the forward and backward attentions in the training stage of decoders. This is implemented as a multitask learning (MTL) framework, in which the forward and backward decoders are trained sharing a single encoder. We aim to regularize the optimization process of one decoder using the loss of the other decoder. The loss function for MTL of the forward decoder is defined as:

$$loss^{(MTL)}_f = \alpha \cdot loss_f + (1 - \alpha) \cdot loss_b, \tag{5.6}$$

where $loss_f$ and $loss_b$ are the original cross-entropy loss functions of the forward and backward decoders, respectively. α is the weight for the main task, which may be set to be larger than 0.5. The shared encoder is trained so that the encoded acoustic representation is expected to be suitable for both of left-to-right and right-to-left decoding.

5.3 Experimental evaluations

We evaluated our methods through speech recognition tasks using the Corpus of Spontaneous Japanese (CSJ) [82]. CSJ includes two distinct subcorpora, namely, CSJ-APS and CSJ-SPS. CSJ-APS consists of academic presentation speeches on several topics such as science, engineering,

[2] This algorithm guarantees that sentences identical to original candidates are also generated due to the existence of ⟨sos⟩ and ⟨eos⟩.

humanities and social science. CSJ-SPS consists of simulated presentation speeches on three general themes. These subsets have their own official test sets, namely, CSJ-TESTSET1 and CSJ-TESTSET3.

A 40-dimensional vector consisting of 40-channel log Mel-scale filterbank (lmfb) outputs is used as acoustic features for attention models. Non-overlapping frame stacking [63] was applied to these features in which we stack and skip three frames to make a new super frame. The acoustic encoders in our attention models consist of 3 or 5-layers of bidirectional LSTMs with 320 cells. Dropout [83] was used for training each LSTM layer with a dropout rate of 0.2. The decoder consists of a 1-layer LSTM with 320 cells and a softmax output layer with the nodes for vocabulary words, $\langle \mathrm{sos} \rangle$, $\langle \mathrm{eos} \rangle$ and $\langle \mathrm{UNK} \rangle$ special token for out-of-vocabulary words. The sizes of word vocabularies for CSJ-APS and CSJ-SPS are 19,146 and 24,826, respectively. We used Adam [57] for optimizing network parameters. We also used gradient clipping with a threshold of 5.0. All network parameters were initialized with random values drawn from a uniform distribution with range (-0.1, 0.1). The input data are sorted by the length of frames before creating minibatches of 30 sentences. While we used Chainer toolkit [96] to train the networks with a 3-layer encoder, we used Pytorch [86] for the larger models with a 5-layer encoder exploiting its advantage of memory efficiency. All experiments were carried out on a system with a 3.60 GHz Intel Xeon and a NVIDIA TITAN X.

5.3.1 Results with small models

First, we show the results of a number of comparative experiments using smaller models with a 3-layer encoder. The beam width N was set to be 4 in all experiments which gave the lowest WERs. Table. 5.1 shows the word error rates (WERs) obtained for two tasks. By comparing the results against the baseline forward and backward decoders, we can find the proposed forward-backward decoder achieves significant improvement in both tasks. Our MTL method yielded small but consistent improvements for all types of decoders. We set the weight α in the MTL to be 0.8. The proposed forward-backward decoding combined with the MTL improved WERs of word-attention models by 12.7 % and 10.3 % relative for CSJ-TESTSET1 and CSJ-TESTSET3, respectively.

We show the error rates of different types (deletion, substitution and insertion) separately in

Table 5.1: ASR performance for two tasks (WER(%))

	CSJ-APS (224 hrs)	CSJ-SPS (251 hrs)
decoder	CSJ-TESTSET1	CSJ-TESTSET3
forward (baseline)	14.51	11.41
+ MTL	14.27	11.21
backward	13.82	11.57
+ MTL	13.58	11.26
forward-backward	**12.92**	**10.38**
+ MTL	**12.67**	**10.24**

Table 5.2: Recognition error rates of different types for CSJ-TESTSET1 (%)

	error type			
decoder	del	sub	ins	total
forward	2.24	9.67	2.60	14.51
+ MTL	2.28	9.71	2.27	14.27
backward	2.47	9.37	1.99	13.82
+ MTL	2.41	9.31	1.86	13.58
forward-backward	2.36	8.70	1.86	12.92
+ MTL	2.24	8.71	1.70	12.67

Table 5.2 in order to investigate in depth the reason for the performance gains from our methods. From this table, we understand that the MTL mainly contributes in reducing insertion errors. On the other hand, the forward-backward decoding improved substitution error rates significantly, in addition to reducing deletion and insertion errors to the level of the lower rate of either decoder.

5.3.2 Results with large models

Here, we present the results using larger models with a 5-layer encoder which have much higher baseline performances. These models were trained using both of CSJ-APS and CSJ-SPS. The

Table 5.3: ASR performance and real time factor (RTF) of proposed method implemented and evaluated with larger models trained using larger data for CSJ-TESTSET1

model		WER(%)	RTF
DNN-HMM + LM		13.62	0.925
phone-CTC + LM		14.15	0.581
forward	N=4	11.27	0.057
	N=2	11.36	0.020
	N=1	11.87	0.010
forward-backward	N=4	**10.17**	**0.119**
	N=2	**10.09**	**0.040**
	N=1	**10.17**	**0.019**

vocabulary size was increased to 34,331 accordingly. We also used scheduled sampling [84] and label smoothing [85] to improve the optimization. A hybrid DNN-HMM model and a phone CTC model were also build using the same training set for comparing their ASR performances and real time factors against our attention-based word models. The DNN-HMM model has seven hidden layers with 2k rectified linear units (ReLUs) and a softmax output layer with 3k nodes. The CTC-based model has 5 layers of bidirectional LSTM with 320 memory cells. For decoding with the hybrid DNN-HMM and the phone CTC, we used the Julius decoder [95] and the EESEN WFST decoder [97], respectively.

The WERs and real time factors (RTF) for CSJ-TESTSET1 are shown in Table 5.3. We also show the results with narrower beam widths than 4 for attention models. The proposed method yielded a further significant performance gain (9.8 % relative) on top of the better baseline model when the beam width was 4. It only took exactly twice as long time as the forward decoder for completing recognition, showing that the Pass 3 in Algorithm 2 required only a negligible time as expected. Interestingly, the proposed method is not affected by a narrow beam width, while the performance of the forward decoder degraded as the beam width became narrower. The results with $N = 1$ show that our algorithm can recover many correct words from only single pair of candidates from two decoders, which the conventional unidirectional decoder can never find even with a wide beam width. The proposed method with $N = 1$ achieved a WER reduction of 25.3 % relative from the DNN-HMM coupled with a trigram language model with a decoding speed faster by a factor of 50.

5.4 Conclusion

We have proposed an effective decoding method by integrating forward and backward attentions for attention-based seq2seq models. The acoustic-to-word model enhanced with the proposed method achieved a WER reduction of 25.3 % relative from a standard hybrid DNN-HMM system with a decoding speed faster by a factor of 50. As future work, we are also interested in constructing an algorithm based on earlier integration of forward and backward decoding as in algorithms based on best-first search [94][95].

Chapter 6

End-to-end generation of written-style transcript from speech for Parliamentary meetings

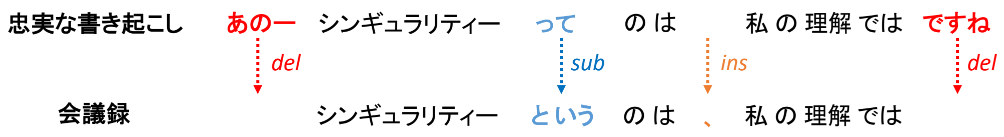
6.1 Introduction

Transcribing and archiving meetings, lectures and presentations is one of the important applications for automatic speech recognition (ASR). In order to make a truly useful archive, we need to not only achieve a low recognition error rate, but also consider the readability of system outputs. Since conventional ASR systems are designed to faithfully reproduce all words actually spoken in an utterance, their outputs are not necessarily easy to read and comprehend due to the existence of spoken language phenomena. For example, spontaneous utterances contain not only fillers and disfluencies, but also redundant and colloquial expressions even when fluently spoken. They are often ungrammatical and lack punctuation marks at all. Consequently, a considerable amount of manual edits are required for making final texts appropriate for a written record from faithful transcripts or ASR results [1].

To address this problem, there have been a number of studies on automatic transformation from spoken to written language. They include disfluency detection and removal [98][99], punctuation insertion [100][101][102], and more general speaking style transformation (SST) [103][104][105][106]. A majority of these works rely on machine learning methods such as noisy channel models, CRFs, SVMs, and deep neural networks. These models are typically trained on annotated texts or a parallel corpus of faithful transcripts and corrected texts independently from speech recognition models. Therefore, speaking style transformation is usually performed as postprocessing of speech recognition and its performance is inevitably affected by errors in the first-path ASR results. Furthermore, since the text-based methods require expensive manual faithful transcripts for training, they are easily prone to the data-sparseness issue.

In this chapter, we propose a novel speech recognition approach that emulates the behavior of a human professional editor by removing fillers and disfluent regions, substituting colloquial expressions with formal ones, inserting punctuation and recovering omitted particles, and performing other types of appropriate corrections, directly using a speech signal as input. Unlike conventional ASR models trained using faithful transcripts as target, in this approach, a Transformer-based sequence-to-sequence model is optimized in an end-to-end manner on the parallel data of speech and corresponding clean transcripts. Therefore, they can naturally avoid the above-mentioned disadvantages of cascade methods that combine independently trained ASR and text-based style transformation models. Because disfluent regions in a spontaneous utterance, which should be corrected in the output, are expected to have more ASR errors than other fluent regions, it may be effective not to explicitly rely on ASR results in generating clean transcripts. Furthermore, the proposed method can perform correction based on acoustic cues present in a speech signal, which was shown to improve the correction performance in [98][105]. In this research, we purposely focus on the specific task of generating the written records of Japanese Parliamentary speeches and provide in-detail evaluation of the proposed method and analysis of the system outputs using this particular task.

This chapter is organized as follows. In section 6.2, we explain why we use Parliamentary meetings as target of this research and provide a detailed analysis on the edit types performed by human editors in creating the official written records of the meetings in the House of Representatives of the Diet. In Section 6.3, we propose a method for generating written-style transcript from speech. Section 6.4 evaluates the proposed method and provides a detailed analysis of the system outputs.



忠実な書き起こし
あのー シンギュラリティー って の は 私 の 理解 では ですね
del
sub
ins
del
会議録
シンギュラリティー という の は 、 私 の 理解 では

Figure 6.1: An example pair of faithful transcript and written record

6.2 Dataset

Figure 6.1 depicts an example of the pair of a faithful transcript and its corresponding clean version from the official record. We can see that a filler word ("e:") and discourse maker (a sentence-end expression "desu ne") were deleted, a colloquial expression was substituted with a formal one (from "tte" to "to iu"), and a comma was inserted at an appropriate position to improve readability.

In this research, we use a corpus consisting of Parliamentary speeches recorded in meetings of the House of Representatives of the Diet (national Parliament) of Japan and their corresponding official written records, mainly for the following reasons.

First, we aim to improve the accuracy and efficiency of the creation process of the written records of Parliamentary meetings through this research. We have been developing the automatic transcription system for Japanese Parliamentary meetings [107][108][109]. This system is in official operation for helping generating the written records of all meetings held in the House of Representatives of the Diet[1] since 2011. In the current scheme, speeches are first faithfully transcribed using a high-performance HMM-based hybrid speech recognition model [19][21], and then professional human editors create the final records by performing necessary corrections to the recognition results. Therefore, if the transcription system can directly generate written-style sentences, the total burden and cost required for creating the final records could be drastically reduced. Furthermore, the inference speed of end-to-end systems used in this research are an order of magnitude faster than the conventional hybrid systems.

Second, the pairs of Parliamentary speeches and their corresponding clean transcripts in the official record provide reasonable examples for investigating the relationship between spoken and written languages. In most Parliamentary meetings, speakers do not just read prepared written

[1]https://www.shugiin.go.jp/internet/itdb_kaigiroku.nsf/html/kaigiroku/kaigi_l.htm

materials but tend to give highly spontaneous utterances. Therefore, as we will see in the next section, there are considerable amount of differences between what are actually spoken in the meetings and clean transcripts in the written records. Furthermore, since it is strictly prohibited for editors to bring any paraphrases in the official records that could potentially alter the intent of the speaker, we can focus on the phenomena purely relevant to the difference between spoken and written languages. Because the official records of Japanese Parliamentary speeches are created by professional editors following very strict and consistent rules, they are appropriate as learning target of machine learning models.

Third, we can use a large amount of reliable paired data of speech and written-style text. The existing researches of speaking-style transformation mostly relied on text-based transformation models trained using faithful transcriptions. Because manually transcribing spontaneous utterances is a very costly process, these models can easily suffer from the data sparsity issue. On the other hand, we aim to directly generate written-style text from speech in an end-to-end manner using paired speech and clean text without using intermediate faithful transcript target.

In this research, we use a dataset collected from the 189-th session of the National Diet held in 2015. The training set used for building all models consists of 194 committee meetings and 14 plenary sessions held until June 2015. The evaluation set consists of 20-hour speech data from 5 committee meetings which we will describe in-detail in the next section.

6.2.1 Edit types in official records of Parliamentary meetings

We manually and faithfully transcribed speech data recorded in five committee meetings from the 189-th session of the National Diet, namely, the 19-th committee meeting on agriculture, forestry and fisheries (229 speaker turns, 22 distinct speakers, 5.5 hours)[2], the 18-th committee meeting on the cabinet (201 speaker turns, 21 distinct speakers, 3.2 hours)[3], the 29-th committee meeting on health, welfare and labour (174 speaker turns, 17 distinct speakers, 4.1 hours)[4], the 4-th committee meeting on consumer affairs (147 speaker turns, 27 distinct speakers, 3.1

[2] https://www.shugiin.go.jp/internet/itdb_kaigirokua.nsf/html/kaigirokua/000918920150625019.htm

[3] https://www.shugiin.go.jp/internet/itdb_kaigirokua.nsf/html/kaigirokua/000218920150708018.htm

[4] https://www.shugiin.go.jp/internet/itdb_kaigirokua.nsf/html/kaigirokua/009718920150708029.htm

hours)[5], the 5-th committee meeting on reconstruction after the great east Japan earthquake (197 speaker turns, 26 distinct speakers, 4.4 hours)[6]. We then annotated the differences between these transcriptions and the corresponding official written records.

Based on these annotations, we analyzed what types of edits or corrections were performed by human editors in creating these official records and how many. We first roughly classified all edits into three major types, namely, corrections by deletional, substitutional, and insertional operations. Then, we reorganized them using finer subclasses from **A** to **M** as follows. Note that in examples of each edit type below, the regions marked with curly brackets {} and parentheses () represent deleted and inserted items, respectively.

Deletion

A. deletion of fillers Word that have morphological forms used only for fillers, namely, 'a:', 'ano:', 'i:', 'u:', 'e:', 'e:to','o:', 'maa', 'N:', and their direct variants such as 'eqto' are categorized as this type.

e.g. 「{ えー } アメリカ議会におきまして」、「{ まあ } しかし { あのー }(、)」

Segments consisting of two or more consecutive filler words are marked as one single filler.

e.g.: 「{ まあ、あのー、おー } 法律の審議でございますので (、)」、「{ えー、ま } 大体もう { あの } 議論は出尽くしたところもありますけれども (、)」

B. deletion of end-of-phrase expressions End-of-phrase expressions that frequently appear in spoken language are deleted.

e.g.: 「正式な場におきまして { ですね }(、) 」、「自民党は { ね }(、) 本当に人情に厚くて { ね }(、)」

C. deletion of earlier parts (reparandum) of repair and repeat Disfluencies overridden in a speech repair are deleted. These disfluent parts often include fillers. Repeated words are also deleted.

[5] https://www.shugiin.go.jp/internet/itdb_kaigirokua.nsf/html/kaigirokua/019718920150709004.htm

[6] https://www.shugiin.go.jp/internet/itdb_kaigirokua.nsf/html/kaigirokua/024218920150709005.htm

e.g.: 「{ 総理もこれまでああー }(大臣もこれまで)」、「{ 質問あの } (決議) に基づきまして (、)」

D. others Words such as 'yahari' and 'mou' used as interjections, and unnecessary subjective or directive words are deleted to make concise sentences.

e.g.: 「{ やはり } 事業を利用する組合員である農業者の利益 (、)」、「そこは残念だと { こう } 思っております (。)」

Substitution

E. correction of wrong particles Particles used in a grammatically wrong manner are replaced with correct ones. The most frequent examples are substitutions from 'ga' to 'ha' (19), from 'ga' to 'wo' (19), from 'no' to 'ni' (12), and from 'ha' to 'ga' (12).

e.g.: 「農業委員の方 { が }(は)(、) 任命制があるので (、)」、「制度自体の大きな枠組みの変更 { が }(を) 検討中だということでございました (。)」

F. correction of colloquial expressions Colloquial expressions in spoken language are substituted with more formal ones.

e.g.: 「{ いろんな }(いろいろな) 声が { えー } あるということ」、「よく承知して { ます }(います){ けども }(けれども)(、)」

G. word reordering Two consecutive words can be replaced with each other form a more readable sentence. This type of correction is seldom performed in recent meetings.

e.g.: 「{ よくないと思うんです一方的すぎると }(一方的すぎると良くないと思うんです)(。)」、「{ 両方出し手と受け手と }(出し手と受け手と両方) 考えてやって」

Non-consecutive words replaced with each other in the written records are not categorized as this edit type but represented as the combination of '**D.** other deletion' and '**M.** other insertion'.

H. correction of disfluencies not repaired by the speaker Disfluencies that were not repaired by the speakers are corrected in the written records by editors. This type of edits also includes semantic corrections.

e.g.: 「{ 組長 }(首長) というのは最終的には」、「{ 農林さんしょう }(農林水産省) の元同僚の皆さんに」

I. other This type corresponds to abbreviation completion or inserting additional expressions that are not present in the original utterances to make more understandable sentences. This type of edits is seldom performed actually.

e.g.: 「第八条 { の四で }(四項の二号で)」、「{ これ }(農業関係は) 二十万人どころじゃないんです (。)」

Insertion

J. comma, K. period Punctuation marks are inserted at appropriate positions to improve readability.

L. recovery of omitted particles Particles are frequently omitted in spoken languages. They are recovered to make grammatical sentences. The most frequent examples are insertions of ’wo’ (604), ’ha’ (549), ’ga’ (204), ’ni’ (151).

e.g.: 「アメリカの議会 (は) この法案の審議をめぐっては」、「ここにいらっしゃる議員 (を) 初め (、)」

M. other The majority of this type of edits are from replacement of non-consecutive words described in type **G**.

e.g.: 「それを { 私は } この場で議論しないのはおかしいと (私は) 思うんですよ (。)」、「{ 比較的 } まだそれでも (、) そういう方の数も (比較的) あるんですが (、)」

6.2.2 Frequency of each edit type

The character-level differences observed in the written records (403.813 characters) in comparison with the faithful transcripts (423,786 characters) are 14,480 substitutions (3.4 %), 19,727 insertions (4.7%), and 39,700 deletions (73,907 characters in total). Figure6.2 shows the proportion of the occurrence counts of each edit type in each committee meeting.

Except for punctuation insertion, about half of all performed corrections were categorized as deletion of fillers. As for corrections with substitutional operations, corrections of colloquial expressions were most frequently observed. These substitutions may require more advanced rules to be performed adequately than straightforward filler-word deletion. We can see that more difficult corrections such as insertion of particles and deletion of disfluencies appeared

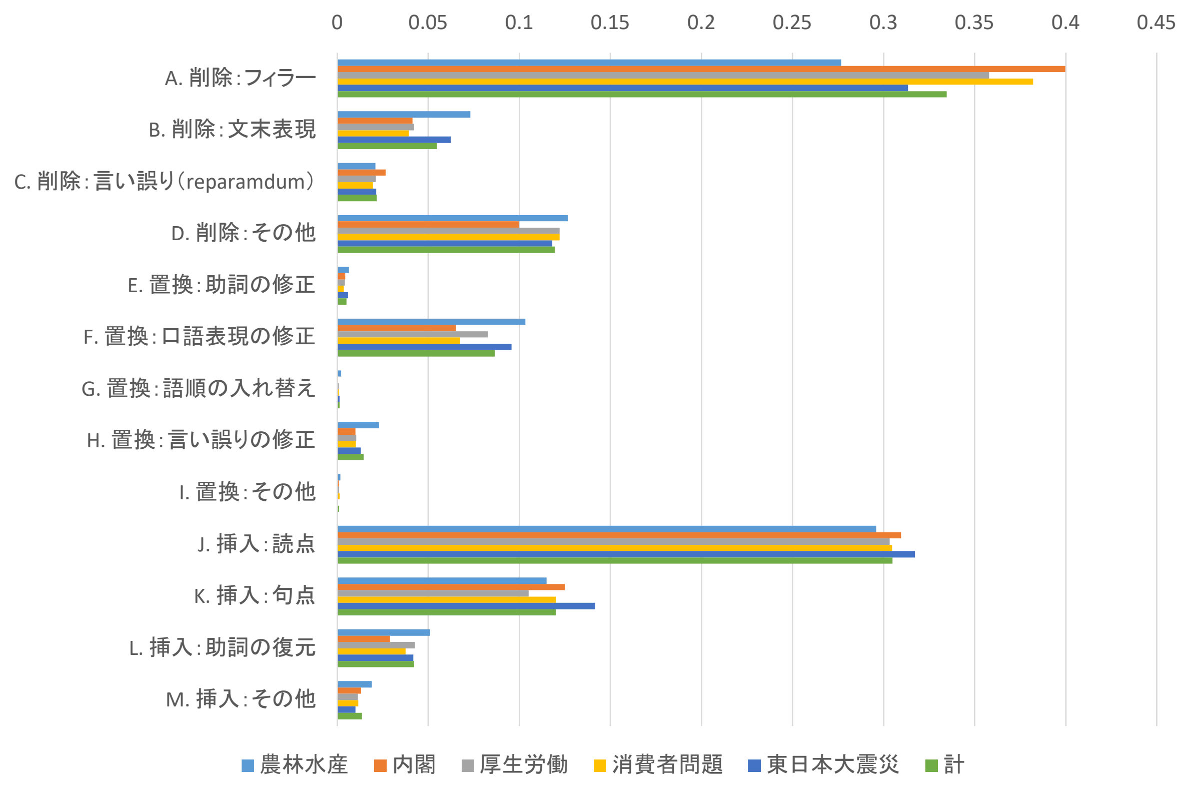


Figure 6.2: Frequency of each edit type in the written record of the National Diet

considerable number of times. On the other hand, non-monotonic corrections such as word reordering are found to be seldom performed actually.

6.3 Proposed method

In the proposed framework, we train a single neural network in an end-to-end fashion using speech as input and written-style text as target. The proposed model directly predicts written-style text from a speech signal in the inference stage. For brevity, we call this approach as *direct generation* of clean texts, the proposed model as a *direct model*, and likewise hereafter.

Advantages of the direct model in comparison with the conventional cascade approach combining ASR and text-based spoken style transformation (SST) are summarized as follows. First, the direct model can perform transformation from spoken to written language using acoustic cues present in a speech signal. By taking this advantage, for example, it can detect and delete disfluent regions in speech or associate pauses to punctuation. Next, the direct approach can

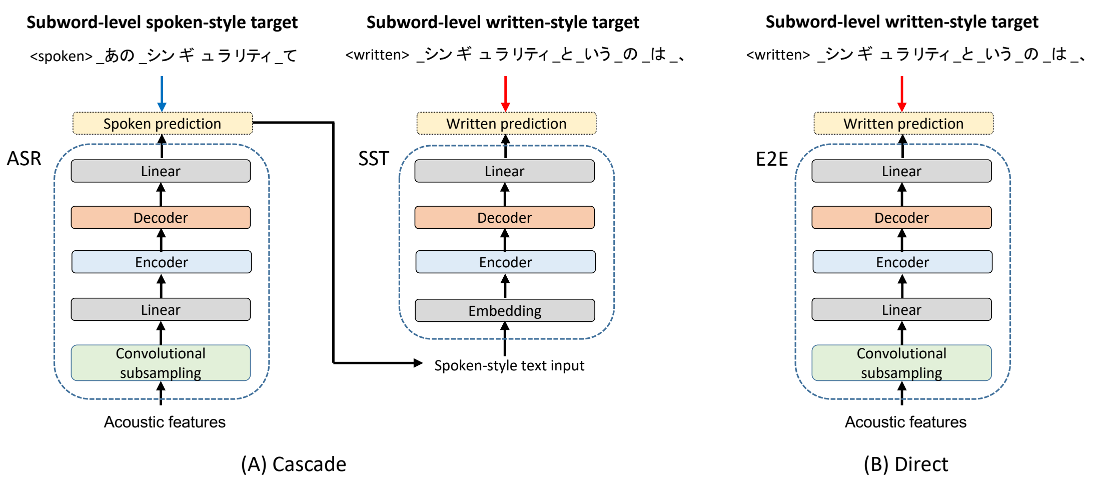

Subword-level spoken-style target
<spoken> _あの _シン ギ ュラリティ _て
Subword-level written-style target
<written> _シン ギ ュラリティ _と _いう _の _は _、
Subword-level written-style target
<written> _シン ギ ュラリティ _と _いう _の _は _、
Spoken prediction
ASR
Linear
Decoder
Encoder
Linear
Convolutional subsampling
Acoustic features
Written prediction
SST
Linear
Decoder
Encoder
Embedding
Spoken-style text input
Written prediction
E2E
Linear
Decoder
Encoder
Linear
Convolutional subsampling
Acoustic features
(A) Cascade
(B) Direct

Figure 6.3: Cascade approach (A) and proposed direct approach (B) for generating written-style transcripts from speech

avoid performance degradation caused by errors from the first-pass ASR results, which is inevitable in the cascade models. Because there is a fundamental problem that speech segments including fillers or disfluencies that should be corrected are expected to be more easily prone to ASR errors, it should be reasonable not to use ASR results at all in performing correction. Furthermore, since we can realize both of ASR and SST within a single network, the architecture of the proposed model is simple and with a compact model size, and thus is easy to adopt in applications. The inference speed is also much faster than the cascade models.

On the other hand, the direct generation requires a model to perform very complicated sequence-to-sequence mappings. For example, it has to insert labels that do not have corresponding acoustic events in speech input (e.g. recovery of omitted particles or insertion of commas), and simultaneously skip speech segments that do not have relevant labels (e.g. deletion of fillers). Therefore, it is considered to be a more difficult task than either of ASR and SST, and requires a very flexible sequence mapping model to be performed appropriately. In this research, we presume that this direct generation is a similar task to end-to-end speech translation [110] and use a Transformer-based label-synchronous seq2seq model to implement it. Figure6.3 depicts the architectural difference between the cascade and proposed direct approaches. In the following sections, we propose techniques to ameliorate these difficulties in the direct generation task.

6.3.1 Improved training strategy using pseudo faithful transcript

As shown in Section 6.2, there are a considerable amount of mismatches between what are actually spoken and clean text target in Parliamentary meetings. Therefore, it can be difficult to train an end-to-end model using only clean text as target. Here we propose to utilize faithful transcripts, which are used in training standard ASR models, to guide the training of the direct model, along with the clean text target. We propose two distinct ways of using faithful transcripts.

Since it is unrealistically expensive to manually transcribe a large amount of spontaneous speech data, we first introduce a method to automatically recover approximate faithful transcripts from the official written records. Then, we describe a procedure to build triplet data of speech, clean text and pseudo transcript. Next, we propose new techniques for improving the training of the direct model based on these triplet data.

Generation of pseudo faithful transcripts

The clean transcript $W_{m,s}$ of a speaker turn s in a meeting m is available for free from the Web, but it is unrealistically expensive to make the manual faithful transcript $V_{m,s}$ for all (m, s). Therefore, we consider to automatically generate the faithful transcript $V_{m,s}$ from the corresponding clean transcript $W_{m,s}$ based on a statistical machine translation framework. Given a written-style text W, we can in principle recover its faithful version of text V using the following Bayes' rule:

$$P(V|W) = P(V) \cdot \frac{P(W|V)}{P(W)} \tag{6.1}$$

However, the transformation from written to spoken language is actually a highly random process. For example, fillers can be potentially inserted to every word boundary in W. Therefore, it is very difficult to uniquely recover the faithful transcript V only from the written-style text W. Here, instead of trying to directly decode the faithful transcript based on the above Bayes' rule, we opt for building a spoken-style statistical language model (LM) $P(V)$ as:

$$P(V) = P(W) \cdot \frac{P(V|W)}{P(W|V)} \tag{6.2}$$

and then performing speech recognition using $P(V)$ to approximately recover V. This LM style transformation is performed by modifying the N-gram counts as follows:

$$Ngram(v_1^n) = Ngram(w_1^n) \cdot \frac{P(v|w)}{P(w|v)} \tag{6.3}$$

where $Ngram(v_1^n)$ and $Ngram(w_1^n)$ are the occurrence counts of each N-gram in the spoken and written-style corpora, respectively. v and w are the unit patterns of transformation in both styles. For example, insertion of filler word 'ano:' is represented as the transformation $\{w = (w_{-1}, w_{+1}) \rightarrow v = (w_{-1}, 'ano:', w_{+1})\}$ in this scheme. We extract all transformation rules between the patterns from a limited amount of parallel data of written-style and faithful transcripts, and find maximum likelihood estimates of the LM parameters $P(v|w)$ and $P(w|v)$. In order to ameliorate the data-sparsity issue, we also use a smoothing technique based on POS (Part-Of-Speech) information. More detailed algorithm of the LM transformation can be found in [111]. The advantage of this technique is that it does not require large paired data unlike deep learning-based methods, and it performs transformation between only specific patterns obtained from the training corpus and thus are guaranteed not to give unexpected transformations harmful for the purpose of recovering the original spoken transcripts such as replacement of content words except for ASR errors.

The detailed procedure for generating the pseudo faithful transcript $\hat{V}_{m,s}$ is as follows.

Step 1 The written-style N-gram LM $P_m(W)$ dependent on meeting m is built on all sentences W_m in the written record of m.

Step 2 The spoken-style N-gram LM $P_m(V)$ is built by applying the LM style transformation described above to the written-style language model $P_m(W)$ from **Step 1**.

Step 3 We build a pronunciation dictionary using the pronunciations of all words appearing in W_m predicted by a morphological analyzer. Using this pronunciation dictionary and the spoken-style language model $P_m(V)$ dependent on meeting m, as well as an HMM-based acoustic model trained on an existing speech corpus, we decode all speech recordings $\boldsymbol{X}_m$ in meeting m. Because $P_m(V)$ is build on only the written record of m and exactly matches the topics discussed in meeting m, it poses a strong constraint in decoding. Furthermore, it can assign appropriate probabilities to spoken-style phenomena such as colloquial expressions. Note that here we use a hybrid ASR system in order to perform speech recognition on the search space constrained by the strong meeting-dependent LM, instead of an end-to-end ASR model that includes an internal language model [112] trained on external data. We use the short pause segmentation algorithm of Julius toolkit [95] for simultaneously performing pause-based speech segmentation and ASR. Because speech recognition using a hybrid system is based on frame-wise predictions, we can use the occurrence times of all words in the recognition result from Julius.

Step 4 The written-style transcript W_m of meeting m is segmented into speaker turn-level texts $W_{m,s}$ according to the speaker information tags available from the official record. We insert a special token into all speaker boundaries and then align the written record and the speech recognition result of the whole meeting. We segment the speech data at the time frames of words aligned with the turn boundary symbol and obtain the speech segment of speaker turn s as $\boldsymbol{X}_{m,s}$.

Step 5 A spoken-style LM $P_{m,s}(V)$ dependent on speaker turn s is built using $W_{m,s}$. This model gives a more strong constraint than $P_m(V)$ built on all texts from m. We decode the speech data $\boldsymbol{X}_{m,s}$ using $P_{m,s}(V)$. We use this very accurate ASR result as the pseudo faithful transcript $\hat{V}_{m,s}$.

Alignment of speech, pseudo faithful transcript and written record

We need triplet data of speech data segmented into manageable lengths $\boldsymbol{X}_{m,s,u}$, their pseudo faithful transcripts $\hat{V}_{m,s,u}$, and the corresponding clean texts in the written record $W_{m,s,u}$ for training the direct model using a standard minibatch-based gradient descent algorithm. We generate these triplets ($\boldsymbol{X}_{m,s,u}$, $\hat{V}_{m,s,u}$, $W_{m,s,u}$) using the following procedure.

Through the short pause segmentation algorithm of Julius, we already obtained the start and end frames of a speech segment $\boldsymbol{X}_{m,s,u}$. Therefore, segmented speech data and their pseudo transcripts $\hat{V}_{m,s,u}$ can be easily obtained. Time information of all words in $\hat{V}_{m,s,u}$ is also available. We align the speaker turn-level pseudo transcript $\hat{V}_{m,s}$ and its clean transcript $W_{m,s}$, and separate $W_{m,s}$ at positions aligned with segment boundary pauses (pauses longer than 200ms) as insertion errors and obtain the written-style text $W_{m,s,u}$ of segment u.

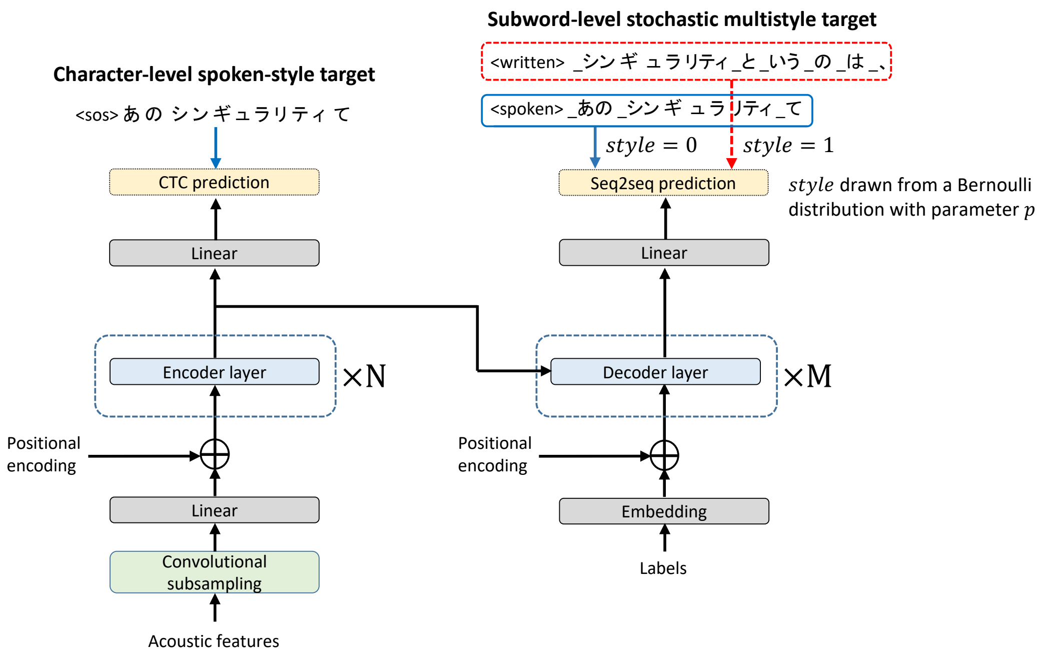

Subword-level stochastic multistyle target
<written> _シン ギ ュラ リティ _と _いう _の _は _、
<spoken> _あの _シン ギ ュラ リティ _て
style = 0
style = 1
Character-level spoken-style target
<sos> あ の シ ン ギ ュ ラ リ テ ィ て
CTC prediction
Seq2seq prediction
style drawn from a Bernoulli distribution with parameter p
Linear
Linear
Encoder layer
×N
Decoder layer
×M
Positional encoding
Positional encoding
Linear
Embedding
Convolutional subsampling
Labels
Acoustic features

Figure 6.4: Encoder multitask learning and decoder multistyle learning using pseudo faithful transcripts

Multitask learning of encoder

In e2e ASR, the role of an acoustic encoder is to remove undesirable deviations caused by noise or speaker characteristics from a speech signal, and extract global information directly useful for predicting linguistic units such as phones [34]. On the other hand, because words and subwords usually consist of multiple syllables and their pronunciations can alter depending on contexts, the correspondence between input speech and labels are very complicated in e2e ASR using grapheme-based output units [67][68]. There are more drastic mismatches between input and labels in speech translation [110] that uses labels of a different language from input speech as target. Therefore, in these models, techniques to help training of the acoustic encoders are usually used.

One such method is to feed lower level labels such as phones or shorter subwords than those used for the final output target, that have a more straightforward correspondence to speech, to the output or intermediate layers of the acoustic encoder as additional learning target [69][113][114]. In e2e speech translation, source language transcriptions are usually used in a multitask training framework [110] or for pretraining of the encoder [115].

Another popular technique is to introduce a subtask based on CTC loss function, in addition to the main task using the cross entropy loss. Because CTC loss enforces a strong constraint of monotonic alignment between input and target, we can more efficiently optimize the encoder by incorporating a CTC-subtask if we have monotonic labels [116][38].

Based on the above considerations, we propose a multitask learning strategy for the acoustic encoder of our direct model. In this strategy, as shown in the encoder part of Figure6.4, we define an ASR subtask designed to predict transcripts as faithful to input speech as possible using the encoder output, besides the main task performed by the decoder that predicts clean text. We use the pseudo faithful transcript $\hat{V}_{m,s,u}$ generated via the procedure in Section 6.3.1 as target of this subtask. We calculate the loss between the target and network prediction using CTC loss function. Because the pseudo transcript $\hat{V}_{m,s,u}$ is recovered by the constrained ASR, it is faithful to speech input. The monotonicity between input and the labels is also guaranteed. The total loss is defined as:

$$loss_{MTL}(\boldsymbol{X}, W, \hat{V}) = \lambda \cdot loss_{ctc}(\boldsymbol{X}, \hat{V}) + (1 - \lambda) \cdot loss_{Transformer}(\boldsymbol{X}, W) \tag{6.4}$$

where λ is a predefined subtask weight.

Multistyle learning of decoder

In the encoder multitask learning described in the previous section, the gradients from the loss calculated using the pseudo transcripts are not propagated to the decoder subnetwork. In this section, we propose a multistyle learning method that is designed for guiding the Transformer decoder using the pseudo transcripts.

This multistyle learning method is formalized using the framework of multilingual e2e ASR [117]. We consider written and spoken languages in Japanese (written record and pseudo faithful transcript) as two different languages, and choose stochastically either of them as target of the decoder for each utterance. We condition the decoder prediction by feeding either of two different start-of-sentence symbols ('`<written>`' and '`<spoken>`' for written and spoken language predictions, respectively). In multilingual ASR, it is known that easy tasks such as English ASR effectively help more challenging tasks such as low-resource language ASR. Likewise, we aim to improve the direct generation of written-style text by incorporating an easier task of standard ASR.

In the inference stage, we feed the decoder '`<written>`' as the start-of-sentence symbol at the initial decoding step. In some applications where faithful outputs are preferable, we instead use '`<spoken>`'.

6.3.2 Automatic speech segmentation based on online punctuation prediction

We need to segment speech data of a whole meeting into manageable lengths (typically, shorter than 20s) prior to performing speech recognition or clean text generation using an e2e model. In this speech segmentation stage, it is desirable if we can segment speech into syntactically and semantically coherent sequential units, which may correspond to sentences in written language. However, manual annotations of such sentence boundaries are not available for a speech corpus in most cases.

In short pause-based speech segmentation, which is usually used in long-form speech recognition, speech is segmented at detected pauses longer than about 200ms. However, in particular in spontaneous utterances, pauses not necessarily coincide with sentence or phrase boundaries. Therefore, pause-based segmentation can produce fragmentation of syntactic structures important for clean text prediction.

On the other hand, if we use longer pauses, pause-based segmentation tends to generate inappropriately long segments in acoustic conditions with low signal-to-noise ratios or utterances with fast speaking rates. An e2e model, in particular an attention-based seq2seq model, is known to drastically degrade the recognition performance for utterances longer than about 20s [118][119]. Therefore, a segmentation strategy that can output very long segments is not desirable. It is also difficult to choose an appropriate threshold of pause length for each evaluation condition [7].

In order to address these problems with the pause-based segmentation, we propose a new segmentation strategy based on punctuation prediction. Punctuation marks (commas and periods)

[7] For example, the 5.5-hour speech recorded in the 19-th committee meeting on agriculture, forestry and fisheries was segmented into 5,062 segments by the short pause segmentation algorithm of Julius using 200ms as the threshold, out of which 1,309 segments were as short as less than 2 seconds. On the other hand, when we used a slightly longer threshold of 300ms, the number of segments shorter than 2 seconds decreased to 340, but instead 166 very long segments (longer than 20 seconds) were generated, which may be very harmful for an seq2seq model.

are expected to appear at a stable rate on the output side, independently to acoustic conditions, speaking styles or speaker characteristics, unlike pauses. Therefore, by considering the positions of punctuation, we can avoid producing too long or short segments. Furthermore, because some coherent sequential information between punctuation marks are at least kept after segmentation, and we can consistently use a reasonable amount of contexts in clean text prediction for each segment.

Online punctuation detection using CTC-based model

The proposed speech segmentation method takes very long speech (typically several hours long) as input. It performs online punctuation prediction from left to right, and segments speech at the speech frames where punctuation marks (commas or periods) are detected. For the punctuation prediction, we use a CTC-based model with a encoder composing of unidirectional LSTM layers. This online CTC-based model is trained using the triplets ($\boldsymbol{X}_{m,s,u}$, $\hat{V}_{m,s,u}$, $W_{m,s,u}$) generated using short pause segmentation algorithm of Julius in Section 6.3.1. In other words, it is built as an online version of the direct model. Note that because an unidirectional model has less expressive power than Transformer-based encoders, it is essential to incorporate multitask learning using pseudo faithful transcripts for convergence as we will see in the experimental section.

The detailed procedure for the speech segmentation method is shown in Algorithm 3. In this algorithm, the unidirectional direct model performs online time-synchronous prediction of clean text, while splitting speech at frames where co-occurrence of a pause with a comma or period is detected. Pauses or non-speech segments can be detected as a sequence of blanks longer than a threshold of N_{blank} [120]. The approximate time of punctuation insertion can be determined by the CTC spike at the corresponding output node. The position of the spike output from an unidirectional CTC model is known to be consistently later than the actual time of appearance of the label. Therefore, the time that is a fixed number of frames (T_{margin}) after the spike is taken as the actual boundary. After resetting the RNN state at the encoder step where the boundary is detected, we continue the time-synchronized clean sentence prediction again.

The reason why we consider not only punctuation but also pauses is as follows. The spikes are not guaranteed to be at the exact time of appearance of corresponding tokens as in the hybrid model. Therefore, punctuation spikes without pauses are very likely to actually fall within the

Algorithm 3 PunctuationBasedSpeechSegmentation($\boldsymbol{X}, N_{blank}, T_{margin}, subsample\_rate$)

```
1: $B$ : set of detected segmentation boundaries
2: $B \Leftarrow \{0\}$, $encoder\_state = 0$, $blank\_count = 0$
3: $\boldsymbol{O} = \mathrm{Subsample}(\boldsymbol{X})$
4: for $t \in [1, 2, ..., T/subsample\_rate]$ do
5:   $encoder\_out, encoder\_state = \mathrm{UnidirectionalRNN}(\boldsymbol{o}_t, encoder\_state)$
6:   $prediction = \arg\max(\mathrm{Linear}(encoder\_out))$
7:   if $prediction = \mathrm{blank}$ then
8:     $blank\_count+ = 1$
9:   else
10:    $previous\_nonblank\_token = prediction$
11:    continue
12:  end if
13:  if $blank\_count > N_{blank}$ then
14:    if $previous\_nonblank\_token \in \{, , .\}$ then
15:      $encoder\_state = 0$
16:      $blank\_count = 0$
17:      $B \Leftarrow t \cdot subsample\_rate - T_{margin}$
18:    end if
19:  end if
20: end for
21: $B \Leftarrow T$
22: return $B$
```

durations of the other subwords before and after them. In addition, without pauses at segment boundaries, speech recognition accuracy is generally degraded due to the lack of clear cues at the beginning and end of sentences. Furthermore, the prediction of punctuation that co-occurs with pauses is considered highly reliable.

6.4 Experimental evaluations

The proposed method was evaluated using a large corpus of meetings from the 189-th session of the National Diet. The training set for each model consisted of 708-hour audio collected from 14 plenary sessions and 194 committee meetings, totaling 208 sessions, held through June 2015. The evaluation set consisted of the five meetings used for the analysis in Section 6.2, namely, the 19-th committee meeting on agriculture, forestry and fisheries (229 speaker turns, 22 distinct speakers, 5.5 hours), the 18-th committee meeting on the cabinet (201 speaker turns, 21 distinct speakers, 3.2 hours), the 29-th committee meeting on health, welfare and labour (174 speaker turns, 17 distinct speakers, 4.1 hours), the 4-th committee meeting on consumer affairs (147 speaker turns, 27 distinct speakers, 3.1 hours), the 5-th committee meeting on reconstruction after the great east Japan earthquake (197 speaker turns, 26 distinct speakers, 4.4 hours). These data were manually annotated with faithful transcriptions as described in section 6.2. We used the 12-th committee meeting on legal affairs (75 speaker turns, 9 distinct speakers, 2.5 hours) as the development set. Various hyperparameters for model optimization and speech segmentation were determined using this development set.

The acoustic features were 80-dimensional log mel filter bank outputs. The analysis window width for acoustic analysis was 25 ms, and a frame shift of 10 ms was used. Each dimension of the features for the training and evaluation data was normalized using the mean and standard deviation of the entire training data. After morphological segmentation using Chasen-2.4.4+Unidic-1.3.9, all text data were tokenized using byte pair encoding (BPE) [121]. The number of distinct subwords is 10k.

For e2e speech recognition and direct clean text generation, we used Transformer-based seq2seq models with the same number of parameters. A 12-layer Transformer was used for the encoder and a 6-layer Transformer for the decoder. The number of heads h, the number of model dimensions d_{model}, and the number of FFN intermediate nodes d_{ff} were set to $h = 4$, $d_{model} =$

256, and $d_{ff} = 2,408$, respectively. The input acoustic feature sequence was subsampled to a length of $1/4$ of the original length using two convolutional layers. Each convolutional layer had 32 output channels, a kernel size of 3, a 2-D CNN with stride 1, and a 2-D pooling layer with stride 2. The CNN output was nonlinearly transformed using the ReLU function [59]. For comparison, e2e speech recognition and direct models based on the CTC loss function were also constructed. The same Transformer encoder as in the seq2seq model was used in the CTC model.

In the cascade method for comparison, the text-based SST model was also implemented as a Transformer-based encoder-decoder model, following a recent cascade approach in speech translation [122]. The encoder and decoder configuration is identical to the e2e speech recognition and direct model described above. The e2e speech recognition and SST models were constructed by a lightly-supervised training framework [123] using pseudo transcriptions described in section 6.3.1. The speech recognition model uses acoustic features as input and pseudo transcription as target, while the SST model was trained with the pseudo transcription as input and the written record as target. For inference, beam search with a beam width of 6 was used for all transformer models, while the greedy search algorithm in [28] was used for the CTC models.

In training the e2e speech recognition and direct models, adaptive Specaugment [124] was used for dynamic data augmentation. All augmentation configurations such as mask probabilities followed the settings in the literature [124]. Models were optimized using the Adam optimizer. All Transformer-based models used the same learning rate scheduling as [39]. That is, the learning rate $lrate(n)$ at n steps was set to

$$lrate(n) = k \cdot d_{model}^{-0.5} \cdot \min(n^{-0.5}, n \cdot warmup\_n^{-1.5}) \tag{6.5}$$

where $warmup\_n = 25,000$ and $k = 4.0$.

In the encoder multitask training, the weight of the ASR subtask is $\lambda = 0.1$. Based on the findings of previous studies [69][114], sub-task targets were character sequences rather than subwords. In the decoder multistyle learning, the parameter p of the Bernoulli distribution used for style selection was set to 0.5. Training was conducted for 100 epochs, and the final network was constructed by averaging the 10 checkpoints that gave the lowest error rates for the development set.

In the hybrid system for comparison, a DNN-HMM acoustic model [21] and a N-gram language model were used. The acoustic model consists of a 7-layer feedforward neural network that identifies 9k classes of triphone states (senone) and an HMM. The acoustic model was trained using the Kaldi toolkit [60]. For a fair comparison, the same 708-hour data recorded in 2015 as the e2e models were used to train this acoustic model. The language model was constructed by applying the spoken language style transformation [111] to the written-style trigram model estimated using written records of meetings held from 2006 to 2015. We used Julius for speech recognition using the hybrid system. The hybrid system was decoded using Julius. In decoding of a whole meeting, as in the creation of the pseudo-transcription in section 6.3.1, speech segmentation and speech recognition were performed simultaneously using the short pause segmentation algorithm (pause length threshold of 200 ms).

In creating the pseudo faithful transcriptions (Section 6.3.1), the conversion rules $P(W|V)$ and $P(V|W)$ were estimated using the parallel data of 666K words of manual transcriptions and corresponding written records of some meetings held in 2003 (mainly from the Budget Committee). The GMM-HMM acoustic model of the hybrid system for this pseudo label generation was trained using pseudo-transcriptions of meetings held from 2009 to 2011 created using the same method as in Section 6.3.1.

Speech data of the five meetings in the evaluation set were segmented using the automatic segmentation method based on CTC (Section 6.3.2), except for the comparison experiment in Section 6.4.4.

We used an AMD-Epyc7262 CPU and an NVidia-RTX-3090 GPU (24G memory) for measuring the inference speed of each method.

6.4.1 Evaluation of baseline ASR models

First, we evaluate the performance of the CTC and Transformer-based e2e ASR models built by the lightly-supervised training using the pseudo faithful transcripts. We show the character error rate calculated using the manual faithful transcripts as reference in Table 6.1 to compare the ASR performance of each model.

Both e2e models achieved lower error rates than the hybrid system using a large statistical language model. The inference speed of the CTC and Transformer-based models on CPU was 270 times and 40 times faster than the hybrid system, respectively. The inference speed was

Table 6.1: Character error rates (%) against faithful transcripts and real time factors of processing time of ASR models

	error type				xRT	
Method	Substitution	Deletion	Insertion	Total	CPU	GPU
Hybrid system	4.3	3.2	3.2	10.7	2.45	-
CTC ASR	3.0	**2.7**	3.6	9.3	**0.009**	**0.003**
Transformer ASR	2.8	3.0	3.2	9.1	0.09	0.04
Transformer ASR + dec. multistyle	**2.7**	2.8	**2.7**	**8.2**	0.09	0.04

even faster on GPU. Note that e2e ASR was performed on speech segments obtained using the CTC-based segmentation, but even when taking this additional processing time into account (xRT=0.03 on CPU, xRT=0.01 on GPU), the e2e models were still several tens of orders faster than the hybrid model.

In the comparison between CTC and Transformer, Transformer yielded a slightly lower error rate than CTC, although it was not as fast as CTC. Furthermore, by incorporating the multistyle learning of the decoder proposed in section 6.3.1, the error rate improved significantly by 0.9 points ('Transformer ASR + dec. multistyle'). This may be because the effect of recognition errors in the labels used in the lightly-supervised learning was mitigated by the use of error-free clean transcript targets. Although the multistyle learning is aimed at improving the direct generation performance, it is noteworthy that this technique was found to be also effective for faithful transcript generation, since some applications require faithful outputs.

We also calculated the error rates for the ASR models using the written records as reference and show them in Table 6.2. To be fair, since the ASR models have no chance to output any punctuation marks, punctuation is excluded from the error rate calculation in this table. Here, insertion errors increased significantly for all models from the results in Table 5.1. The performance of the hybrid system improved by 7.5 points in absolute by simply removing lexical fillers defined in the pronunciation dictionary. This is consistent with the analysis in Chapter 2, which showed that almost half of the corrections in the written records were removal of fillers.

Table 6.2: Character error rates (%) against written records and real time factors of processing time of ASR models. Punctuation marks were excluded from error rate calculation

	Error type				xRT	
Method	Substitution	Deletion	Insertion	Total	CPU	GPU
Hybrid system	4.5	2.7	14.1	21.3	2.45	-
+ filler word removal	4.1	3.1	6.6	13.8	2.45	-
CTC ASR	3.3	2.9	15.2	21.4	**0.009**	**0.003**
Transformer ASR	3.0	2.6	14.2	19.7	0.09	0.04
+ Transformer SST	3.0	3.8	**3.2**	9.9	0.18	0.08
Transformer ASR + dec. multistyle	**2.9**	**2.3**	13.6	18.8	0.09	0.04
+ Transformer SST	3.0	3.4	3.3	**9.7**	0.18	0.08

The Transformer-based ASR seq2seq model remained superior to the CTC. Furthermore, the Transformer-based text-based SST applied to the first-path ASR outputs significantly reduced the number of insertion errors. This improvement from SST was significantly larger than the removal of lexical fillers in the hybrid system, which suggests that the Transformer-based seq2seq transformation allows for advanced editing operations beyond simple rule-based deletions. The performance was further improved by using the multistyle model in the first-path e2e ASR, achieving the lowest overall error rate (9.7 %).

To summarize, the models built with lightly-supervised learning achieved consistently high levels of speech recognition accuracies. From this, we understand that the proposed method in section 6.3.1 is effective enough to produce reliable transcriptions.

6.4.2 Comparison between cascade and direct models

Here, we compare the performance of the cascade and proposed direct models in generating written-style transcripts. The error rates for these models calculated using the written records as reference are shown in Table 6.3. Here, punctuation prediction errors are also included in the

Table 6.3: Character error rates (%) against written records and real time factors of processing time of direct models

	Error type				xRT	
Method	Substitution	Deletion	Insertion	Total	CPU	GPU
Transformer cascade	3.3	4.5	3.5	11.4	0.18	0.08
Transformer cascade + dec. multistyle	3.3	4.1	3.6	11.0	0.18	0.08
CTC direct	(did not converge)					
CTC direct + enc. multitask	**2.4**	4.2	3.2	9.8	**0.009**	**0.003**
Uni. CTC direct	(did not converge)					
Uni. CTC direct + enc. multitask	3.6	5.2	3.5	12.3	0.03	0.01
Transformer direct	2.8	3.6	3.2	9.6	0.09	0.04
Transformer direct + enc. multitask	**2.1**	**2.8**	3.2	8.2	0.09	0.04
Transformer direct + dec. multistyle	2.2	2.9	3.0	8.2	0.09	0.04
Transformer direct + both	**2.1**	3.0	**2.7**	**7.8**	0.09	0.04

error rates. The cascade model shown in this table is identical to the one that yielded the result of the combination of Transformer ASR and SST in Table 6.2.

The direct model based on Transformer ('Transformer direct') achieved a 1.4 points lower error rate than the best cascade model ('Transformer cascade + dec. multistyle'). In addition, its inference speed was twice as fast as the cascade model. This confirms the superiority of the direct approach as described at the beginning part of Chapter 4. The CTC direct model ('CTC direct'), on the other hand, did not converge until the 100-th epoch at all and failed to output any meaningful results. This clearly shows that the Transformer-based seq2seq model is more suitable than the CTC-based one for the task of predicting clean transcripts, which requires many deletion, substitution, and insertion operations, in addition to performing accurate speech recognition.

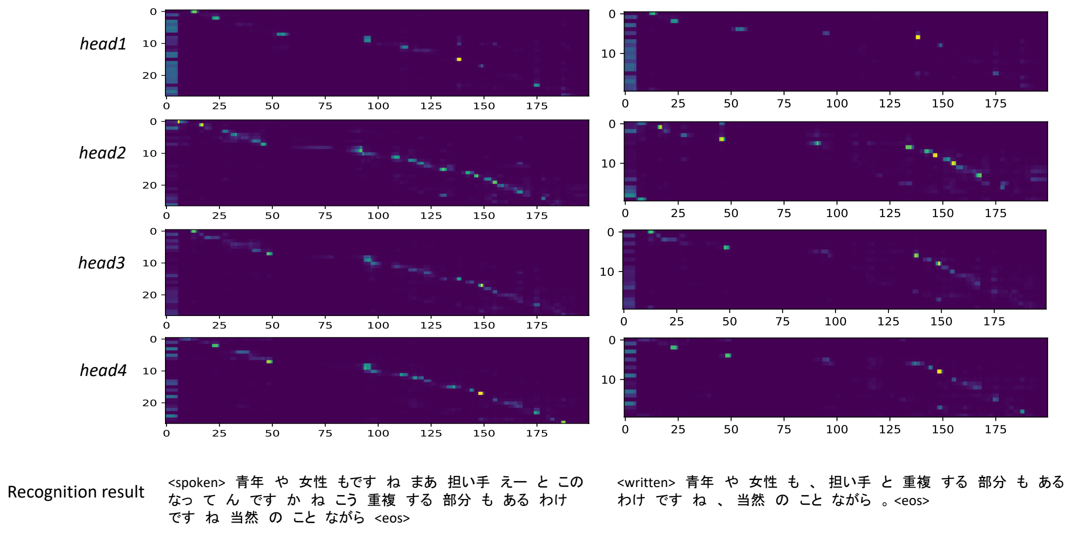
Table 6.4: Effects of subtask targets in multitask learning (character error rates (%))

	Error type			
Subtask target	Substitution	Deletion	Insertion	Total
Faithful, character	**2.1**	**2.8**	3.2	**8.2**
Faithful, subword	2.6	3.2	**3.0**	8.7
Clean, character	2.9	4.2	4.2	11.3

6.4.3 Effectiveness of training strategies using pseudo faithful transcripts

Next, we evaluate the proposed methods for guiding the training of the direct models using pseudo transcription (Sections 5.3.1). The multitask learning of acoustic encoder ('Transformer direct + enc. multitask') improved the direct generation performance significantly by 1.4 points compared to the baseline model trained using only clean texts as target. The CTC-based direct model this time successfully converged and achieved a reasonable performance of 9.8 %. However, it still fell short of the Transformer baseline built without the multitask learning. Similarly, the unidirectional CTC model, which we built for the task of online speech segmentation (Section 5.3.2), converged for the first time with the encoder multitask learning. These results indicate that the use of speech-faithful labels plays a very important role in improving the direct prediction of clean transcript from speech. It is important to note that these significant improvements were obtained using automatically generated labels based on a statistical machine translation framework, rather than manually annotated reference labels.

The impact of different types of target labels used in the multitask learning is shown in Table 6.4. We can see that the direct generation performance was significantly better when using characters than subword-level labels as the subtask target. This may be due to the improved efficiency of the CTC subtask by using output units with a larger number of training examples per class. When clean sentences were used as the target of the subtask, the performance was rather significantly decreased. This result, together with the fact that the CTC-based direct model did not converge at all with the clean target alone, clearly indicates that the CTC loss function is not suitable for handling the clean sentence target.



<spoken> 青年 や 女性 もです ね まあ 担い手 えー と この なっ て ん です か ね こう 重複 する 部分 も ある わけ です ね 当然 の こと ながら <eos>

<written> 青年 や 女性 も 、 担い手 と 重複 する 部分 も ある わけ です ね 、 当然 の こと ながら 。<eos>

Figure 6.5: Example visualizations of the top layer cross-attention weights Transformer-based multistyle model

Next, we evaluate the effect of the decoder multistyle learning using a probabilistic combination of pseudo-transcription and clean text targets. As shown in Table 5.3, this technique achieved an improvement of 1.4 points from the baseline. The error rate was further reduced by 0.4 points by using the encoder multitask learning and the decoder multistyle learning at the same time.

To understand the behavior of the direct model in predicting clean sentences, we show visualized examples of top layer cross-attention weights from the multistyle model in Figure 6.5. The attention weights on the left column are obtained by conditioning the multistyle model with the `<spoken>` tag and forcing it operate as an ASR model, and those on the right are obtained by conditioning with the `<written>` tag. When looking at the attention weights in the right column, we can see that two disfluent regions, "ですねまあ" from around frame 100 to 200, and "この何て言うんですかね" from frame 400 to 520, are skipped appropriately. In addition, the particle "と" and "担い手" surrounded by them were output correctly.

6.4.4 Effectiveness of speech segmentation based on punctuation detection

We here evaluate the speech segmentation method using punctuation as a cue (Section 5.3.2). Specifically, we compared three different CTC-based segmentation strategies. The first one

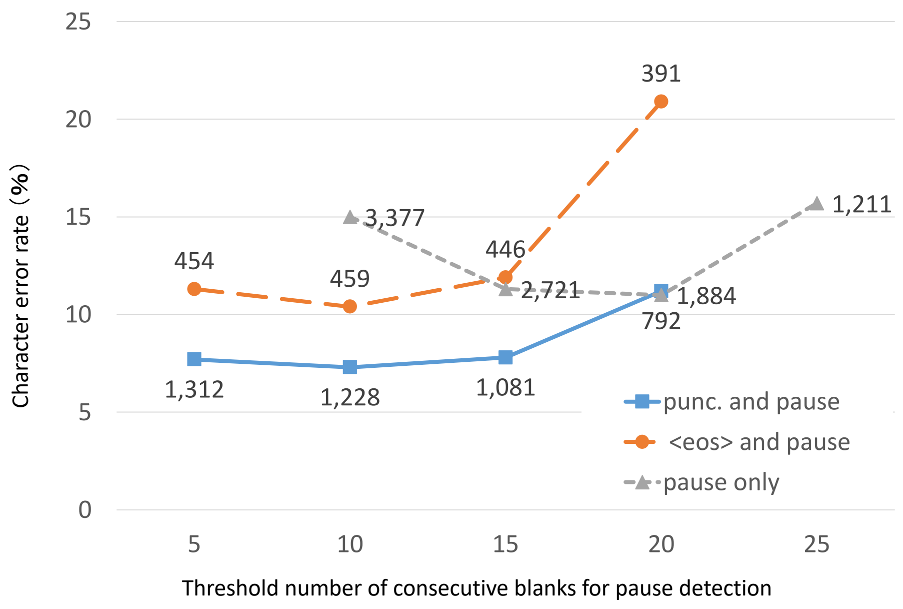


Figure 6.6: Effect of the threshold numbers of consecutive blanks in CTC-based speech segmentation

segments speech data at a punctuation mark that co-occurs with a pause detected as consecutive blank frames longer than a threshold (proposed method), while the second one segments when a pause and an end-of-sentence symbol (`<eos>`) co-occur. The third one segments speech at all detected pauses.

The error rates for the development set with different threshold numbers of consecutive blanks (N_{blank}) in Algorithm 3 are shown in Figure 6.6. The number of segments generated by the combination of each method and threshold is also shown above or below each data point. For all experiments in this figure, the Transformer-based direct model trained with the encoder multitask learning was used for clean text generation.

From this figure, we can see that the proposed punctuation-based method gives consistently high accuracies up to about 15 blanks. The third strategy using only pauses as boundary cues yielded higher error rates than the proposed method, and its threshold dependence was also higher than that of the proposed method. In the setting where the proposed method gave the lowest error rate ($N_{blank} = 10$), the development data was divided into 1,228 segments, whereas the pause-based method generated a similar number of segments at $N_{blank} = 25$. However, the latter gave a significantly higher error rate (15.7 %) than the proposed method (7.3 %). The standard deviation of segment length was 4.95 seconds for the proposed method, while it was

Table 6.5: Comparison of speech segmentation methods in training and evaluation sets

Training set / Evaluation set	short pause (17.973)	CTC-based segmentation using punctuation (10,083)
shot pause (616,047)	9.8	22.2
punctuation (442,816)	10.2	**8.2**

7.06 seconds and there was a large variation in segment lengths for the latter method. The second strategy using the end-of-sentence symbol `<eos>` tends to consistently produce segments that are 20 to 25 seconds long, independent of sentence or speaker boundaries. This may be bacause segments of these lengths rarely appeared in the training data.

Next, we evaluated the clean text generation performance with the combinations of segmentation methods in the training and evaluation sets. For the short pause-based segmentation, we used Julius' short pause segmentation algorithm for both training and evaluation data. The threshold was set to 200 ms, which is commonly used in speech recognition. As for punctuation-based segmentation, we segmented speech in the training data at only detected pauses (detected by Julius) that were aligned with punctuation marks in the written records, instead of segmenting at all pauses. The evaluation data were segmented using the CTC-based segmentation method. We used optimal hyperparameters obtained from the experiment in Figure 1 using the development set ($N_{blank} = 10$、$T_{margin} = 30$).

The error rates of the evaluation data for the four combinations of segmentation methods on the training and evaluation sets are shown in Table6.5. By considering punctuation in both stages, the error rate is 1.6 points lower than the short pause segmentation. When the training data was segmented using pauses and the evaluation data was segmented by punctuation, deletion errors were significantly increased. This may be due to the high proportion of very short segments in the training set segmented by short pauses and relatively long segments were not correctly recognized during evaluation.

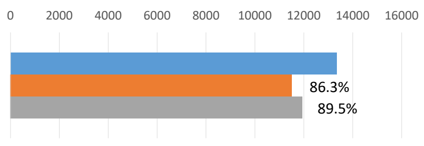
As shown in Table6.3 ('uni. CTC direct + enc. multitask'), the performance of the unidirectional CTC in direct clean text prediction is not as good as other models based on the Transformer, because it does not use bidirectional sequence information. However, if we focus on the punctuation insertion performance, the unidirectional model achieved a high precision rate of 0.934 for periods and 0.811 for commas, which was likely to be sufficient for speech segmentation.

6.4.5 Error analisys

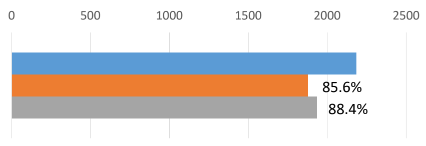
This section evaluates the degree of achievement of the proposed direct generation model for each edit type in comparison with human editors. Figure6.7 shows the number of corrections, except for punctuation insertion, correctly performed by the cascade and direct models, as well as their recall rates. The cascade model used in this comparison is the one that combines the multistyle model and the Transformer-based SST postprocessing. The direct model was constructed using both of the encoder multitask learning and decoder multistyle learning.

When looking at the results for the deletional corrections, we see that both models achieved high recalls for the deletion of lexical fillers and end-of-phrase expressions. On the other hand, for deletion of reparandums in repair and repeat, the performance gap between the cascade and direct models was found to be very large (52.4 vs 79.9 %). This suggests that acoustic information is useful for identifying non-fluent reparandums. The cascade model is likely to have been unable to recover the ASR errors at the 2nd path SST. The performance for “other deletion” type turned out to be commonly very low. This may be due to the fact that there are many examples that require high-level judgment in this type, such as the deletion of contextually unnecessary subjects and directives.

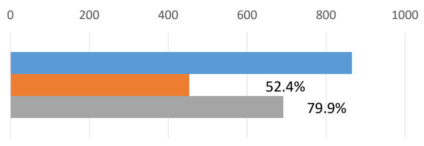
Among the substitutional operations, correction of particles and disfluencies resulted in very low recall rates. These corrections are not likely to be possible with the seq2seq model alone, since the frequencies of their occurrences are very low in the corpus and performing semantic corrections is necessary here. In correction of particles, there were many errors that were perceived to be faithful to the actual pronunciation, especially in the correction of "が" to "は" and "が" to "を". Most of the examples of word reordering were not corrected appropriately. The cascade model that does not use acoustic information succeeded in some straightforward examples, e.g., correction from "今審議がなされている" to "審議が今なされている". As a result, the cascade model gave a higher recall rate than the direct model in this edit type. In contrast, both models showed very high recall rates in correction of colloquial expressions, since many examples in this edit type actually can be performed as straightforward paraphrases, such



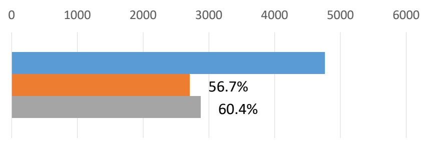
(a) Deletion: fillers



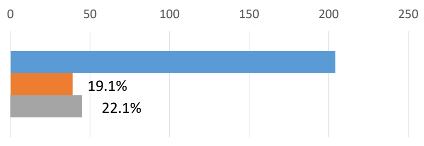
(b) Deletion: end-of-phrase expressions



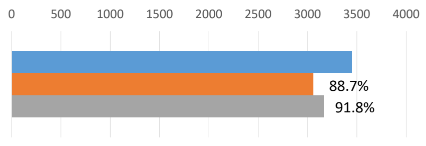
(c) Deletion: disfluencies



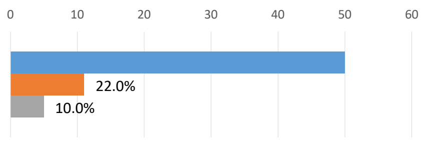
(d) Deletion: other



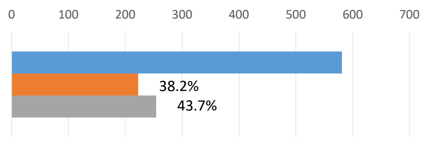
(e) Substitution: corrections of particles



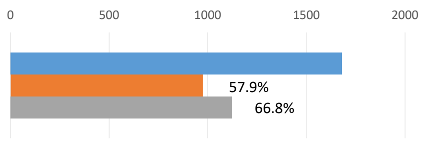
(f) Substitution: corrections of colloquial expressions



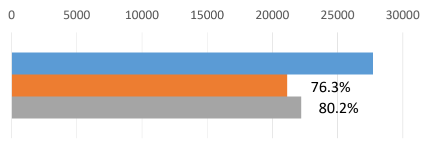
(g) Substitution: word reorder



(h) Substitution: corrections of disfluencies



(i) Insertion: recovery of omitted particles



(j) All (except for punctuation insertion)

🟦 Human editor 🟧 Cascade model ⬜ Direct model

Figure 6.7: The numbers of correctly performed corrections by the cascade and direct models in each edit type, and their recall rates to those by human transcribers

Table 6.6: Performance of punctuation insertion

	comma			period		
Method	Recall	Precision	F-value	Recall	Precision	F-value
Cascade model	**0.79**	**0.74**	**0.76**	0.90	0.81	0.86
Direct model	**0.79**	0.73	**0.76**	**0.94**	**0.84**	**0.88**

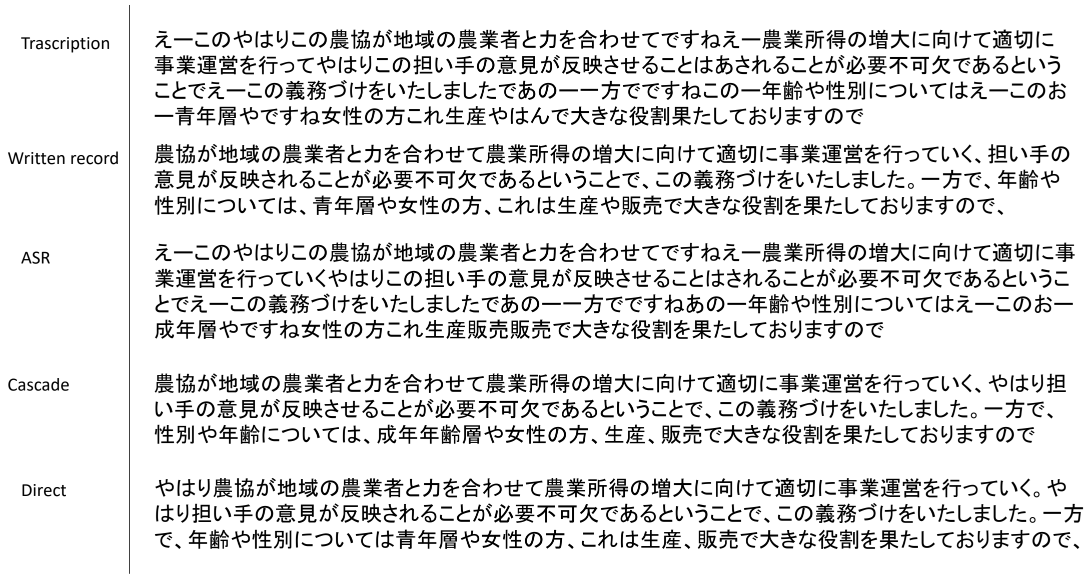
as "いろんな" to "いろいろな" and "やつ" to "もの".

Among the insertion operations, particle insertion was reproducible at a reasonable level. In the cascade model, there were many ASR errors before and after the omission of particles, and the performance gap between the direct model and the cascade model was large.

To summarize, the direct model performed significantly better than the cascade model for almost all edit types. The performance gap was particularly large for advanced corrections such as deletion of reparandums and restoration of particles. Overall, the direct model was able to reproduce 80.2 percent of all manual edits in the official record.

Finally, we evaluate the performance of punctuation insertion in the system outputs. Table6.6 shows recall, precision, and F-values for punctuation insertion with the cascade model and the direct model. There was no difference in comma insertion performance between the cascade and direct models. For period insertion, the direct model showed significantly higher performance. This may be due to that the acoustic information was particularly effective for periods, which correlate better with pauses.

Figure6.8 shows an example of the manual transcription, written record text, system outputs from the ASR, cascade, and direct models. From this example, we understand that the ASR result is difficult to read even though its accuracy is very high, and the proposed method appropriately improves readability. We can also see that the direct model successfully performed some advanced edits such as restoring the particle in "これは" and deleting only the repurundum part “させることはあ” in "反映させることはあされることが". On the other hand, some issues can be seen, such as the failure to remove the word "やはり" at the beginning of the input.



Trascription	えーこのやはりこの農協が地域の農業者と力を合わせてですねえー農業所得の増大に向けて適切に事業運営を行ってやはりこの担い手の意見が反映させることはあされることが必要不可欠であるということでえーこの義務づけをいたしましたであのーー方でですねこの一年齢や性別についてはえーこのおー青年層やですね女性の方これ生産やはんで大きな役割果たしておりますので
Written record	農協が地域の農業者と力を合わせて農業所得の増大に向けて適切に事業運営を行っていく、担い手の意見が反映されることが必要不可欠であるということで、この義務づけをいたしました。一方で、年齢や性別については、青年層や女性の方、これは生産や販売で大きな役割を果たしておりますので、
ASR	えーこのやはりこの農協が地域の農業者と力を合わせてですねえー農業所得の増大に向けて適切に事業運営を行っていくやはりこの担い手の意見が反映させることはされることが必要不可欠であるということでえーこの義務づけをいたしましたであのーー方でですねあの一年齢や性別についてはえーこのおー成年層やですね女性の方これ生産販売販売で大きな役割を果たしておりますので
Cascade	農協が地域の農業者と力を合わせて農業所得の増大に向けて適切に事業運営を行っていく、やはり担い手の意見が反映させることが必要不可欠であるということで、この義務づけをいたしました。一方で、性別や年齢については、成年年齢層や女性の方、生産、販売で大きな役割を果たしておりますので
Direct	やはり農協が地域の農業者と力を合わせて農業所得の増大に向けて適切に事業運営を行っていく。やはり担い手の意見が反映されることが必要不可欠であるということで、この義務づけをいたしました。一方で、年齢や性別については青年層や女性の方、これは生産、販売で大きな役割を果たしておりますので、

Figure 6.8: Examples of reference faithful transcript, written record, and system outputs

6.5 Conclusion

In this study, we proposed a novel speech recognition framework that outputs highly readable written style sentences directly from speech. Experiments on large data sets consisting of pairs of audio recordings of the Parliamentary meetings and their corresponding written records have shown that the proposed method is significantly better than the cascade method that combines speech recognition and text-based style transformation in clean text generation. Among all edit types, in the deletion of reparandums and the restoration of particles, which are considered to be greatly affected by speech recognition errors, and the insertion of punctuation marks, for which acoustic information is considered to be important, the proposed model was particularly effective compared to the cascade approach.

We also proposed a new techniques for guiding the training of the direct model using pseudo faithful transcriptions automatically generated based on a statistical machine translation framework, namely, encoder multitask learning and decoder multistyle learning. The performance of the direct model was significantly improved with these techniques. Furthermore, we proposed a new speech segmentation method using a CTC-based online punctuation prediction model. It was shown to be more effective than conventional methods based on short pauses for the direct generation task.

On the other hand, we observed some of the correction items resulted in low recall rates, and there still remains a room for further model improvement. Correction and restoration of particles can be improved by simply increasing the number of training examples or by post-editing such as deep learning-based tagging methods [125] using a large language model. It is also important to evaluate the generalizability of the proposed approach using Parliamentary speech corpora in other languages, such as the European Parliament [126] and the Icelandic Parliament [126].

There are many application domains where there is potential demand for automatic clean text generation such as lectures and presentations [82]. Since large and reliable annotations of written texts are usually not available for these domains, knowledge transfer from the direct generation model built using the Parliamentary meetings is another important direction to be considered in future work. One promising strategy for adapting the Parliamentary model to another application domain, where we have only faithful labels for each utterance, is as follows: Firstly, annotate a certain part of the new dataset with written-style transcripts at an acceptable cost. Then, fine-tune the Parliamentary model using not only the annotated utterances but also the remaining larger part with only faithful labels based on the multistyle learning framework.

Chapter 7

Conclusion

In this thesis, we have presented four distinct novel techniques, all of which are dedicated to compensating the undesirable biases we encounter when transcribing and archiving speech recordings from real meetings and presentations.

The cycle GAN-based acoustic feature transformation presented in Chapter 2 aims to improve the ASR performance under acoustically unknown conditions. We evaluated this approach on two different cross-domain tasks, namely, speech enhancement and speaking style transformation. The results from the latter experiment demonstrated that the proposed model can learn a reliable mapping even in a very challenging task where we cannot generate paired source and target domain features via simulation. The effectiveness of this cycle GAN-based feature adaptation was also confirmed in later studies. Hosseini et al. [127] demonstrated its significant effectiveness in gender adaptation for English speech recognition. Matsuura et al. [128] use the cycle GAN-based nonparallel mapping for speaker adaptation in order to address the speaker sparsity problem in low-resourced speech recognition and drastically improved the transcription performance for unknown speakers compared to other existing approaches such as multilingual training and unsupervised adaptation using pseudo labels.

Archiving recordings of real spoken communication requires expertise on the specific topic in a target domain. Correctly transcribing technical terms and named entities is very important for later meaningful indexing and retrieval, which may not be covered well by a limited amount of annotated real speech data available for training an e2e model. The TTS-based data augmentation in Chapter 3 generates synthetic paired examples using only text resources relevant to a target domain. We showed that this drastically improved the ASR performance on a number of different cross-domain ASR scenarios. In the academic presentation transcription experiment, it successfully reproduced technical terms that did not appear in the real speech training data. Recently, this TTS-based data augmentation has already become the de facto standard for text-based e2e model adaptation due to its consistent effectiveness. However, it requires a considerable amount of computation for data synthesis and e2e model retraining, which should be addressed in future work.

The bidirectional decoding mechanism described in Chapter 4, on the other hand, compensates the myopic bias in autoregressive label-synchronous speech recognition. It significantly improved the attention-based ASR in presentation speech recognition experiments by incorporating target-side global information. The use of bidirectional information in autoregressive ASR was also confirmed in other recent studies using external bidirectional language models such as BERT for rescoring and knowledge distillation [129][130][131]. The forward-backward decoder was also shown to be effective in clean transcript generation in [132].

We addressed the problem of the readability of ASR outputs for spontaneous utterances via e2e clean transcript generation in Chapter 5. We proposed new techniques for guiding the training of the direct model using pseudo faithful transcriptions automatically generated based on a statistical machine translation framework, namely, encoder multitask learning and decoder multistyle learning. The proposed e2e approach successfully reproduced more than 80% of all manual corrections. On the other hand, we observed some of the correction items resulted in low reproducibility, and there still remains a room for further model improvement. Since large and reliable annotations of written texts are not available for many application domains, knowledge transfer from the reliable Parliamentary direct model to those target domains is another important direction in future work.

Acknowledgment

I would like to thank my supervisor Prof. Tatsuya Kawahara for his guidance and advice. I also like to thank Prof. Shinsuke Mori and Prof. Takayuki Ito for their valuable comments and feedback that greately improved the quality of this manuscript. Dr. Shinsuke Sakai co-authored all the conference papers that this thesis is based on. Prof. Yuya Akita kindly provided me with the language modeling tools I used for pseudo labeling of Parliamentary speeches in Section 6. I am greateful for their helpful contributions. I also have to thank Dr. Koji Inoue, Mayumi Abe, Dr. Yoshiaki Bando, Kazuki Shimada, Kohei Matsuura, Hayato Futami, and everyone else in the speech and audio processing laboratory at Kyoto University for their generous help in my research projects relevant to this thesis.

Bibliography

[1] D. Jones, F. Wolf, E. Gibson, E. Williams, E. Fedorenko, D. Reynolds, and M. Zissman, "Measuring the readability of automatic speech-to-text transcripts," Eurospeech, pp.1585–1588, 2003.

[2] F. Jelinek, "Continuous Speech Recognition by Statistical Methods," Proceedings of the IEEE, pp.532–556, 1976.

[3] R. Schwartz, Y. Chow, O. Kimball, S.Roucos, M. Krasner, and J. Makhoul, "Context-dependent Modeling for Acoustic Phonetic Recognition of Continuous Speech," ICASSP, 1985.

[4] L.R. Rabiner, S. Levinson, and M. Sondhi, "On the Use of Hidden Markov Models for Speaker-Independent Recognition of Isolated Words From a Medium-Size Vocabulary," AT&T BELL LABORATORIES TECHNICAL JOURNAL, pp.627–642, 1984.

[5] L.R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," Proceedings of the IEEE, pp.257–286, 1989.

[6] B.H. Juang, S.E. Levinson, and M. Sondhi, "Maximum Likelihood Estimation for Multivariate Mixture Observations of Markov Chains," IEEE TRANSACTIONSON INFORMATIONTHEORY, pp.307–309, 1986.

[7] L. Bahl, P. de Souza, P. Gopalakrishnan, D. Nahamoo, and M. Picheny, "Decision Trees for Phonological Rules in Continuous Speech," ICASSP, pp.185–188, 1991.

[8] S. Young, J. Odell, and P. Woodlang, "Tree-Based State Tying for High Accurach Acoustic Modelling," HLT, pp.307–312, 1994.

[9] L.Lee and R.C.Rose, “Speaker normalization using efficient frequency warping procedures,” ICASSP, pp.353–356, 1996.

[10] S.Wegmann, D.McAllaster, J.Orloff, and B.Peskin, “Speaker normalization on conversational telephone speech,” ICASSP, pp.339–342, 1996.

[11] C.J.Leggetter and P.C.Woodland, “Maximum likelihood linear regression for speaker adaptation of continuous density hidden markov models,” Computer Speech and Language, pp.171–185, 1995.

[12] L.R. Bahl, P.F. Brown, P.V. de Souza, and R. Mercer, “Maximum Mutual Information Estimation of Hidden Markov Parameters for Speech Recognition,” ICASSP, pp.49–52, 1986.

[13] D. Povey and P. Woodland, “Minimum phone error and i-smoothing for improved discriminative training,” ICASSP, pp.105–108, 2002.

[14] D. Povey, B. Kingsbury, L. Mangu, G. Saon, H. Soltau, and G. Zweig, “Fmpe: discriminatively trained features for speech recognition,” ICASSP, pp.961–964, 2005.

[15] R. Kneser and H. Ney, “Improved backing-off for m-gram language modeling.,” ICASSP, pp.181–184, 1986.

[16] M. Mohri, F. Pereira, and M. Riley, “Weighted Finite-State Transducers in Speech Recognition,” Computer Speech and Language, pp.97–106, 2000.

[17] F. Seide, G. Li, and D. Yu, “Conversational Speech Transcription Using Context-Dependent Deep Neural Networks,” Interspeech, pp.437–440, 2011.

[18] G.E.Dahl, D.Yu, L.Deng, and A.Acero, “Context-dependent pre-trained deep neural networks for large vocabulary speech recognition,” IEEE Trans. Audio, Speech, & Language Proc., vol.20, no.1, pp.30–42, 2012.

[19] A. Mohamed, G. Dahl, and G. Hinton, “Acoustic modelling using deep belief networks,” IEEE Trans. Audio, Speech, & Language Proc., vol.20, no.1, pp.14–22, 2012.

[20] G.E. Hinton, S. Osindero, and Y.W. Teh, “Fast Learning Algorithm for Deep Belief Nets,” Neural Computation, pp.1527–1554, 2006.

[21] G. Hinton, L. Deng, D. Yu, G. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, “Deep neural networks for acoustic modeling in speech recognition,” IEEE Signal Processing Magazine, vol.29, no.6, pp.82–97, 2012.

[22] S.Hochreiter and J.Schmidhuber, “Long Short-Term Memory,” Neural Computation, vol.9, no.8, pp.1735–1780, 1997.

[23] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K.J. Lang, “Phoneme Recognition Using Time-Delay Neural Networks,” IEEE Transactions on Acoustics, Speech, and Signal Processing, pp.328–339, 1989.

[24] K. Veseĺ, A. Ghoshal, L. Burget, and D. Povey, “Sequence-discriminative training of deep neural networks,” Interspeech, pp.2345–2349, 2013.

[25] B. Kingsbury, “Lattice-based optimization of sequence classification criteria for neural-network acoustic modeling,” ICASSP, pp.3761–3764, 2009.

[26] D. Povey, V. Peddinti, D. Galvez, P. Ghahremani, V. Manohar, X. Na, Y. Wang, and S. Khudanpur, “Purely sequencetrained neural networks for ASR based on lattice-free MMI,” Interspeech, pp.2751–2755, 2016.

[27] T. Mikorov, M. Karafiát, L. Burget, J.H. Černocký, and S. Khudanpur, “Recurrent neural network based language model,” Interspeech, pp.1045–1048, 2010.

[28] A. Graves, S. Fernandez, F. Gomez, and J. Schmidhuber, “Connectionist temporal classification : Labelling unsegmented sequence data with recurrent neural networks,” Proc. of the 23st International Conference on Machine Learning, pp.369–376, 2006.

[29] A. Graves, “Sequence transduction with recurrent neural networks,” LCML, pp.4945–4949, 2012.

[30] J. Chorowski, D. Bahdanau, K. Cho, and Y. Bengio, “End-to-end continuous speech recognition using attention-based recurrent NN: first results,” NIPS: Workshop Deep Learning and Representation Learning Work- shop, 2014.

[31] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, “Listen, attend and spell: A neural network for large vocabulary conversational speech recognition,” ICASSP, pp.4960–4964, 2016.

[32] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. ukasz Kaiser, and I. Polosukhin, “Attention is all you need,” 31st Conference on Neural Information Processing Systems (NIPS 2017), 2017.

[33] A. van den Oord, Y. Li, and O. Vinyals, “Representation Learning with Contrastive Predictive Coding,” arXiv:1807.03748, 2018.

[34] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, “wav2vec 2.0: A framework for self-supervised learning of speech representations,” Advances in Neural Information Processing Systems, pp.12449–12460, Curran Associates, Inc., 2020.

[35] W.N. Hsu, B. Bolte, Y.H.H. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed, “HuBERT: Self-Supervised Speech Representation Learning by Masked Prediction of Hidden Units,” IEEE/ACM Transactions on Audio, Speech, and Language Processing, pp.3451–3460, 2021.

[36] C. Gulcehre, O. Firat, K. Xu, K. Cho, L. Barrault, H.C. Lin, F. Bougares, H. Schwenk, and Y. Bengio, “On using monolingual corpora in neural machine translation,” arXiv:1503.03535, 2015.

[37] J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, “Attention-based models for speech recognition,” Advances in Neural Information Processing Systems (NIPS), pp.577–585, 2015.

[38] S. Karita, N.E.Y. Soplin, S. Watanabe, M. Delcroix, A. Ogawa, and T. Nakatani, “Improving transformer-based end-to-end speech recognition with connectionist temporal classification and language model integration,” Interspeech, pp.1408–1412, 2019.

[39] L. Dong, S. Xu, and B. Xu, “Speech-transformer: A no-recurrence sequence-to-sequence model for speech recognition,” ICASSP, pp.5884–5888, 2018.

[40] S. Karita, N. Chen, T. Hayashi, T. Hori, H. Inaguma, Z. Jiang, M. Someki, N. Enrique, Y. Soplin, R. Yamamoto, X. Wang, S. Watanabe, T. Yoshimura, and W. Zhang, "A comparative study on transformer vs rnn in speech applications," ASRU, pp.449–456, 2019.

[41] X. Feng, Y. Zhang, and J. Glass, "Speech Feature Denoising and Dereverberation via Deep Autoencoders for Noisy Reverberant Speech Recognition," Proc. ICASSP, pp.1778–1782, 2014.

[42] F. Weninger, S. Watanabe, Y. Tachioka, and B. Schuller, "Deep Reccurent De-noising Auto-encoder and Blind De-reverberation for Reverberated Speech Recognition," Proc. ICASSP, pp.4656–4660, 2014.

[43] X.Lu, Y.Tsao, S.Matsuda, and C.Hori, "Speech enhancement based on deep denoising autoencoder," Interspeech, pp.436–440, 2013.

[44] J. Du, Q. Wang, T. Gao, Y. Xu, L. Dai, and C.H. Lee, "Robust speech recognition with speech enhanced deep neural networks," Interspeech, pp.616–620, 2014.

[45] M.Mimura, S.Sakai, and T.Kawahara, "Reverberant speech recognition combining deep neural networks and deep autoencoders augmented with a phone-class feature," EURASIP journal on Advances in Signal Processing, 2015.

[46] K.Kinoshita, M.Delcroix, T.Yoshioka, T.Nakatani, E.Habets, R.Haeb-Umbach, V.Leutnant, A.Sehr, W.Kellermann, R.Maas, S.Gannot, and B.Raj, "The reverb challenge: A common evaluation framework for dereverberation and recognition of reverberant speech," Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA-13), 2013.

[47] J.Barker, R.Marxer, E.Vincent, and S.Watanabe, "The third chime speech separation and recognition challenge: Dataset, task and baselines," Proc. ASRU, 2015.

[48] I.J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in Advances in Neural Information Processing Systems 27, pp.2672–2680, 2014.

[49] J.Y. Zhu, T. Park, P. Isola, and A.A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," arXiv preprint arXiv:1703.10593, 2017.

[50] L. Deng, M. Seltzer, D. Yu, A. Acero, A. rahman Mohamed, and G. Hinton, "Binary coding of speech spectrograms using a deep auto-encoder," Interspeech, pp.1692–1695, 2010.

[51] T.Ishii, H.Komiyama, T.Shinozaki, Y.Horiuchi, and S.Kuroiwa, "Reverberant speech recognition based on denoising autoencoder," Interspeech, pp.3512–3516, 2013.

[52] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," ECCV, pp.694–711, Springer, 2016.

[53] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," CVPR, pp.770–778, 2016.

[54] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Instance normalization: The missing ingredient for fast stylization.," arXiv preprint arXiv:1607.08022, 2016.

[55] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, "Improved training of Wasserstein GANs," arXiv preprint arXiv:1704.00028, 2017.

[56] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein GAN," arXiv preprint arXiv:1701.07875, 2017.

[57] D.P. Kingma and J. Ba, "Adam: A method for stochastic optimization," ICLR, 2015.

[58] O. Abdel-Hamid, A. rahman Mohamed, H. Jiang, L. Deng, G. Penn, and D. Yu, "Convolutional neural networks for speech recognition," IEEE Trans. Audio, Speech & Language Process., vol.22, no.10, pp.1533–1545, 2015.

[59] V. Nair and G.E. Hinton, "Rectified Linear Units Improve Restricted Boltzmann Machines," Proceedings of ICML, pp.807–814, 2010.

[60] 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," Proc. ASRU, pp.1–4, 2011.

[61] A. Odena, C. Olah, and J. Shlens, “Conditional image synthesis with auxiliary classifier GANs,” arXiv preprint arXiv:1610.09585, 2016.

[62] A. Graves and N. Jaitly, “Towards end-to-end speech recognition with recurrent neural networks,” Proc. of the 31st International Conference on Machine Learning, pp.1764–1772, 2014.

[63] H. Sak, A. Senior, K. Rao, and F. Beaufays, “Fast and accurate recurrent neural network acoustic models for speech recognition,” arXiv preprint arXiv:1607.06947, 2015.

[64] H. Sak, F. de Chaumont Quitry, T. Sainath, and K. Rao, “Acoustic modelling with CD-CTC-SMBR LSTM RNNs,” ASRU, pp.604–609, 2015.

[65] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio, “End-to-end attention-based large vocabulary speech recognition,” ICASSP, pp.4945–4949, 2016.

[66] A. Graves, A. rahman Mohamed, and G. Hinton, “Speech recognition with deep recurrent neural networks,” ICASSP, pp.6645–6649, 2013.

[67] K. Audhkhasi, B. Ramabhadran, G. Saon, M. Picheny, and D. Nahamoo, “Direct acoustics-to-word models for English conversational speech recognition,” Interspeech, pp.959–963, 2017.

[68] H. Soltau, H. Liao, and H. Sak, “Neural speech recognizer: Acoustic-to-word LSTM model for large vocabulary speech recognition,” Interspeech, pp.3707–3711, 2017.

[69] S. Ueno, H. Inaguma, M. Mimura, and T. Kawahara, “Acoustic-to-word attention-based model complemented with character-level CTC-based model,” ICASSP, 2018.

[70] M. Mimura, S. Sakai, and T. Kawahara, “Forward-backward attention decoder,” Interspeech, pp.2232–2236, 2018.

[71] Y. Wang, R. Skerry-Ryan, D. Stanton, Y. Wu, R.J. Weiss, N. Jaitly, Z. Yang, Y. Xiao, Z. Chen, S. Bengio, Q. Le, Y. Agiomyrgiannakis, R. Clark, and R.A. Saurous, “Tacotron: Towards end-to-end speech synthesis,” Interspeech, pp.4006–4010, 2017.

[72] J. Shen, R. Pang, R.J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Skerry-Ryan, R.A. Saurous, Y. Agiomyrgiannakis, and Y. Wu, “Natural tts synthesis by conditioning wavenet on mel spectrogram predictions,” ICASSP, pp.4779–4783, 2018.

[73] W. Ping, K. Peng, A. Gibiansky, S.Ö. Arik, A. Kannan, and S. Narang, “Deep voice 3: Scaling text-to-speech with convolutional sequence learning,” ICLR, 2018.

[74] J. Sotelo, S. Mehri, K. Kumar, J.F. Santos, K. Kastner, A. Courville, and Y. Bengio, “Char2wav: End-to-end speech synthesis,” ICLR, 2017.

[75] S. Ioffe and C. Szegedy, “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift,” Proceedings of ICML, pp.448–456, 2015.

[76] J. Li, G. Ye, R. Zhao, J. Droppo, and Y. Gong, “Acoustic-to-word model without OOV,” ASRU, 2017.

[77] Z. Chen, Q. Liu, H. Li, and K. Yu, “On modular training of neural acoustics-to-word model for LVCSR,” ICASSP, pp.4754–4758, 2018.

[78] S. Ueno, T. Moriya, M. Mimura, S. Sakai, Y. Shinohara, Y. Yamaguchi, Y. Aono, and T. Kawahara, “Encoder transfer for attention-based acoustic-to-word speech recognition,” Interspeech, 2018.

[79] J. Chorowski and N. Jaitly, “Towards better decoding and language model integration in sequence to sequence models,” Interspeech, pp.523–527, 2017.

[80] T. Hori, S. Watanabe, and J. Hershey, “Multi-level language modeling and decoding for open vocabulary end-to-end speech recognition,” ASRU, 2017.

[81] A. Kannan, Y. Wu, P. Nguyen, T.N. Sainath, Z. Chen, and R. Prabhavalkar, “An analysis of incorporating an external language model into a sequence-to-sequence model,” ICASSP, pp.5824–5828, 2018.

[82] K. Maekawa, “Corpus of Spontaneous Japanese: Its design and evaluation,” Proc. ISCA & IEEE Workshop on Spontaneous Speech Processing and Recognition, pp.7–12, 2003.

[83] N.Srivastava, G.Hinton, A.Krizhevsky, I.Sutskever, and R.Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” Journal of Machine Learning Research, vol.15, pp.1929–1958, 2014.

[84] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, “Scheduled sampling for sequence prediction with recurrent neural networks,” NIPS, pp.1171–1179, 2015.

[85] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the inception architecture for computer vision,” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016.

[86] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, “Automatic differentiation in pytorch,” 2017.

[87] R. Sonobe, S. Takamichi, and H. Saruwatari, “JSUT corpus: free large-scale Japanese speech corpus for end-to-end speech synthesis,” arXiv preprint arXiv:1711.00354, 2017.

[88] D. Krueger, T. Maharaj, J. Kramár, M. Pezeshki, N. Ballas, N.R. Ke, A. Goyal, Y. Bengio, A. Courville, and C. Pal, “Zoneout: Regularizing RNNs by randomly preserving hidden activations,” LCLR, 2017.

[89] A. Tjandra, S. Sakti, and S. Nakamura, “Listening while speaking: Speech chain by deep learning,” ASRU, 2017.

[90] J.G. Fiscus, “A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (ROVER),” ASRU, 1997.

[91] L. Mangu, E. Brill, and A. Stolcke, “Finding consensus in speech recognition: word error minimization and other applications of confusion networks,” vol.14, no.1, pp.373–400, 2000.

[92] T. Menne, J. Heymann, A. Alexandridis, K. Irie, A. Zeyer, M. Kitza, P. Golik, I. Kulikov, L. Drude, R. Schlüter, H. Ney, R. Häb-Umbach, and A. Mouchtaris, “The RWTH/UPB/FORTH system combination for the 4th CHiME challenge evaluation,” 2016.

[93] S. Audtin, R. Schwartz, and P. Placeway, “The forward-backward search algorithm,” ICASSP, 1991.

[94] F.K. Soong and E.F. Huang, “A tree-trellis based fast search for finding the N best sentence hypotheses in continuous speech recognition,” ICSLP, 1990.

[95] A. Lee, T. Kawahara, and K. Shikano, “Julius : an open source real-time large vocabulary recognition engine,” EUROSPEECH, pp.1691–1694,.

[96] S. Tokui, K. Oono, S. Hido, and J. Clayton, “Chainer: a next-generation open source framework for deep learning,” Proceedings of Workshop on Machine Learning Systems (LearningSys) in The Twenty-ninth Annual Conference on Neural Information Processing Systems (NIPS), 2015.

[97] Y. Miao, M. Gowayyed, and F. Metze, “EESEN: End-to-end speech recognition using deep RNN models and WFST-based decoding,” Proc. ASRU, pp.167–174, 2015.

[98] Y. Liu, E. Shriberg, A. Stolcke, D. Hillard, M. Ostendorf, , and M. Harper, “Enriching speech recognition with automatic de- tection of sentence boundaries and disfluencies,” IEEE Trans. Audio, Speech & Language Process., vol.14, pp.1526–1540, 2006.

[99] J. Yeh and C. Wu, “Edit disfluency detection and correc- tion using a cleanup language model and an alignment model,” IEEE Trans. Audio, Speech & Language Process., vol.14, pp.1574–1583, 2006.

[100] M. Paulik, S. Rao, I. Lane, S. Vogel, and T. Schultz, “Sentence segmentation and punctuation recovery for spoken language translation,” Proc. ICASSP, 2008.

[101] A. Gravano, M. Jansche, and M. Bacchiani, “Restoring punctuation and capitalization in transcribed speech,” Proc. ICASSP, pp.4741–4744, 2009.

[102] Y. Akita and T. Kawahara, “Automatic comma insertion of lecture transcripts based on multiple annotations,” Interspeech, pp.2889–2892, 2011.

[103] T. Hori, D. Willett, and Y. Minami, “Paraphrasing spontaneous speech using weighted finite-state transducers,” ISCA & IEEE Workshop on Spontaneous Speech Processing and Recognition, 2013.

[104] K. Shitaoka, H. Nanjo, and T. Kawahara, “Automatic transformation of lecture transcription into document style using statistical framework,” Interspeech, pp.2169–2172, 2004.

[105] G. Neubig, Y. Akita, S. Mori, and T. Kawahara, “A monotonic statistical machine translation approach to speaking style transformation,” Computer Speech and Language, pp.349–370, 2012.

[106] R. Sproat and N. Jaitly, “An rnn model of text normalization,” Interspeech, pp.754–757, 2017.

[107] Y. Akita, M. Mimura, and T. Kawahara, “Automatic transcription system for meetings of the japanese national congress,” Interspeech, pp.84–87, 2009.

[108] T. Kawahara, “Transcription system using automatic speech recognition for the Japanese Parliament (Diet),” AAAI/IAAI, pp.2224–2228, 2012.

[109] T. Kawahara, “Captioning software using automatic speech recognition for online lectures,” The Journal of Professional Reporting and Transcription (Tiro), 2021.

[110] R.J. Weiss, J. Chorowski, N. Jaitly, Y. Wu, and Z. Chen, “Sequence-to-Sequence Models Can Directly Translate Foreign Speech,” Interspeech, pp.2625–2629, 2017.

[111] Y. Akita and T. Kawahara, “Topic-independent speaking-style transformation of language model for spontaneous speech recognition,” ICASSP, pp.33–36, 2007.

[112] E. McDermott, H. Sak, and E. Variani, “A Density Ratio Approach to Language Model Fusion in End-to-end Automatic Speech Recognition,” ASRU, pp.434–441, 2019.

[113] Y. Higuchi, K. Karube, T. Ogawa, and T. Kobayashi, “Hierarchical conditional end-to-end asr with ctc and multi-granular subword units,” ICASSP, 2022.

[114] R. Sanabria and F. Metze, “Hierarchical multitask learning with ctc,” SLT, pp.485–490, 2018.

[115] A. Bérard, L. Besacier, A.C. Kocabiyikoglu, and O. Pietquin, “End-to-end automatic speech translation of audiobooks,” ICASSP, pp.6224–6228, 2018.

[116] S. Kim, T. Hori, and S. Watanabe, “Joint ctc- attention based end-to-end speech recognition using multi-task learning,” ICASSP, pp.4835–4839, 2017.

[117] S. Watanabe, T. Hori, and J.R. Hershey, “Language independent end-to-end architecture for joint language identification and speech recognition,” ASRU, pp.265–271, 2017.

[118] C.C. Chiu, W. Han, Y. Zhang, R. Pang, S. Kishchenko, P. Nguyen, A. Narayanan, H. Liao, S. Zhang, A. Kannan, R. Prabhavalkar, Z. Chen, T. Sainath, and Y. Wu, “A comparison of end-to-end models for long-form speech recognition,” ASRU, pp.889–896, 2019.

[119] J. Pan, T. Lei, K. Kim, K. Han, and S. Watanabe, “Sru++: Pioneering fast recurrence with attention for speech recognition,” ICASSP, 2022.

[120] T. Yoshimura, T. Hayashi, K. Takeda, and S. Watanabe, “End-to-end automatic speech recognition integrated with ctc-based voice activity detection,” ICASSP, pp.6999–7003, 2020.

[121] R. Sennrich, B. Haddow, and A. Birch, “Neural machine translation of rare words with subword units,” Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, pp.1715–1725, 2016.

[122] L. Bentivogli, M. Cettolo, M. Gaido, A. Karakanta, A. Martinelli, M. Negri, and M. Turchi, “Cascade versus direct speech translation: Do the differences still make a difference?,” Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, p.2873âŁ“2887, 2021.

[123] L. Lamel, J. Gauvain, and G. Adda, “Investigating lightly supervised acoustic model training,” ICASSP, pp.477–480, 2001.

[124] D.S. Park, Y. Zhang, C.C. Chiu, Y. Chen, B. Li, W. Chan, Q.V. Le, and Y. Wu, “Specaugment on large scale datasets,” ICASSP, pp.6879–6883, 2020.

[125] E. Malmi, S. Krause, S. Rothe, D. Mirylenka, and A. Severyn, “Encode, tag, realize: High-precision text editing,” Proceedings of the 2019 Conference on Empirical Methods

in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, pp.5054–5065, 2019.

[126] S. Steingr'ımsson, S. Barkarson, and G.T. Örnólfsson, "IGC-Parl: Icelandic Corpus of Parliamentary Proceedings," LREC, pp.11–17, 2020.

[127] E. Hosseini-Asl, Y. Zhou, C. Xiong, and R. Socher, "Augmented cyclic adversarial learning for low resource domain adaptation," ICLR, 2019.

[128] K. Matsuura, M. Mimura, S. Sakai, and T. Kawahara, "Generative Adversarial Training Data Adaptation for Very Low-resource Automatic Speech Recognition," Interspeech, pp.2737–2741, 2020.

[129] Y. Bai, J. Yi, J. Tao, Z. Wen, Z. Tian, and S. Zhang, "Integrating Knowledge into End-to-End Speech Recognition from External Text-Only Data," IEEE/ACM Transactions on Audio, Speech, and Language Processing, pp.1340–1351, 2021.

[130] H. Futami, H. Inaguma, S. Ueno, M. Mimura, S. Sakai, and T. Kawahara, "Distilling the knowledge of BERT for sequence- to-sequence ASR," Interspeech, pp.3635–3639, 2020.

[131] H. Futami, H. Inaguma, M. Mimura, S. Sakai, and T. Kawahara, "ASR rescoring and confidence estimation with ELECTRA," ASRU, pp.380–387, 2021.

[132] M.Mimura, S.Shinsuke, and T.Kawahara, "An end-to-end model from speech to clean transcript for parliamentary meetings," APSIPA ASC, 2021.

List of publications

Chapter 3

Refereed publications

[1] M. Mimura, S. Sakai, T. Kawahara, “Cross-domain Speech Recognition Using Nonparallel Corpora with Cycle-consistent Adversarial Networks,” Proc. IEEE ASRU 2017, pp.134–140, December 2017.

Chapter 4

Refereed publications

[1] M. Mimura, S. Sakai, T. Kawahara, “Leveraging sequence-to-sequence speech synthesis for enhancing acoustic-to-word speech recognition,” Proc. IEEE SLT 2018, pp.477–484, December 2018.

Chapter 5

Refereed publications

[1] M. Mimura, S. Sakai, T. Kawahara, “Forward-Backward Attention Decoder,” Proc. Interspeech 2018, pp.2232–2236, August 2018.

Chapter 6

Refereed publications

[1] M. Mimura, S. Sakai, and T. Kawahara, “An end-to-end model from speech to clean transcript for Parliamentary meetings,” Proc. APSIPA ASC 2021, pp.465–470, December 2021.

[2] 三村正人 河原達也, “国会会議録のための音声から書き言葉への end-to-end 変換,” 自然言語処理 (掲載見込)