End-to-end ASR to jointly predict transcriptions and linguistic annotations

Motoi Omachi and **Yuya Fujita**
Yahoo Japan Corporation

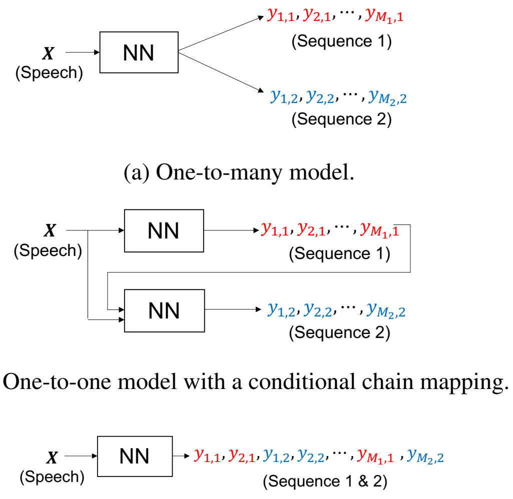
Shinji Watanabe and **Matthew Wiesner**
Center for Language and Speech Processing
Jhons Hopkins University

Abstract

We propose a Transformer-based sequence-to-sequence model for automatic speech recognition (ASR) capable of simultaneously transcribing and annotating audio with linguistic information such as phonemic transcripts or part-of-speech (POS) tags. Since linguistic information is important in natural language processing (NLP), the proposed ASR is especially useful for speech interface applications, including spoken dialogue systems and speech translation, which combine ASR and NLP. To produce linguistic annotations, we train the ASR system using modified training targets: each grapheme or multi-grapheme unit in the target transcript is followed by an aligned phoneme sequence and/or POS tag. Since our method has access to the underlying audio data, we can estimate linguistic annotations more accurately than pipeline approaches in which NLP-based methods are applied to a hypothesized ASR transcript. Experimental results on Japanese and English datasets show that the proposed ASR system is capable of simultaneously producing high-quality transcriptions and linguistic annotations.

1 Introduction

End-to-end automatic speech recognition (E2E ASR), which transcribes speech using a single neural network (NN), has recently gained traction (Graves and Jaitly, 2014; Chorowski et al., 2015; Chan et al., 2016; Graves, 2012; Dong et al., 2018). Existing E2E ASR models generate audio transcripts by sequentially producing likely graphemes, or multi-graphemic units, from which lexical items of a language can be recovered. However, other linguistic annotations such as phonemic transcripts, part-of-speech (POS) tags, or word boundaries, help understand the underlying audio characteristics (Simonnet et al., 2017). Such linguistic annotations are especially important in natural language processing (NLP) tasks done on audio data, including spoken dialogue systems (Jurafsky and Martin, 2008). This study aims to endow existing E2E ASR models with the ability to produce such linguistic annotations.


X
(Speech)
NN
$y_{1,1}, y_{2,1}, \cdots, y_{M_1,1}$
(Sequence 1)
$y_{1,2}, y_{2,2}, \cdots, y_{M_2,2}$
(Sequence 2)

(a) One-to-many model.

X
(Speech)
NN
$y_{1,1}, y_{2,1}, \cdots, y_{M_1,1}$
(Sequence 1)
NN
$y_{1,2}, y_{2,2}, \cdots, y_{M_2,2}$
(Sequence 2)

(b) One-to-one model with a conditional chain mapping.

X
(Speech)
NN
$y_{1,1}, y_{2,1}, y_{1,2}, y_{2,2}, \cdots, y_{M_1,1}, y_{M_2,2}$
(Sequence 1 & 2)

(c) One-to-one model with a single sequence.

Figure 1: Example of E2E ASR predicting two types of sequences. $y_{m,l}$ denotes the m-th token of the l-th type of the sequence.

Prior work explored using E2E ASR systems to predict multiple kinds of labels. Fig. 1 shows a diagram of these systems. These approaches use one of the following models: a one-to-many (O2M) model (Kubo and Bacchiani, 2020; Ueno et al., 2018; Gowda et al., 2019; Sanabria and Metze, 2018; Adams et al., 2019), a one-to-one (O2O) model with a conditional chain mapping (Shi et al., 2020), or an O2O model with a single sequence (Audhkhasi et al., 2018; Ghannay et al., 2018; Shafey et al., 2019; Yadav et al., 2020).

In O2M models shown in Fig. 1(a), a multi-task objective is used in which an extra branch is tasked with estimating the secondary label sequence. For example, in (Kubo and Bacchiani, 2020), the phonemic transcript is produced in addition to the graphemic transcript. The O2M model can estimate each sequence more accurately than

Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 1861–1871
June 6–11, 2021. ©2021 Association for Computational Linguistics

separate models responsible for producing phonemic and graphemic transcripts independently. We can implement this approach with less effort by attaching multiple loss functions to the base architecture. However, this O2M model does not explicitly consider dependencies between phonemic and graphemic transcripts. Furthermore, aligning phoneme and grapheme sub-sequences requires additional post-processing based on time alignment or alignment across the multiple sequences during inference. Performance of downstream NLP tasks built on top of ASR outputs will suffer if this post-processing fails to generate alignment.

Fig. 1(b) shows an O2O model with a conditional chain mapping. This method for multiple sequence modeling has been applied to dialog modeling (Liang et al., 2020), speaker diarization (Fujita et al., 2020a), and multi-speaker ASR (Shi et al., 2020). Unlike the O2M model, this model can predict a variable number of output sequences while explicitly considering dependencies between the multiple sequences based on the probabilistic chain rule. However, modeling these inter-sequence dependencies requires more complicated neural architectures, and alignment of the sequences still requires post-processing during inference.

Another option for using O2O models is to output multiple sequences as a *single* sequence instead of using conditional chain mapping, as shown in Fig. 1(c). For example, in (Audhkhasi et al., 2018), the O2O model produces word transcripts by first generating a word's constituent graphemes followed by the word itself. Another application, explored in (Shafey et al., 2019) used the O2O model to produce graphemes followed by speaker role. This approach is the simplest to implement because we can reuse the neural network architecture used to produce the primary sequence to sequence mapping to produce the secondary label sequence (e.g., connectionist temporal classification (CTC) based systems). In contrast to the previous two approaches, the O2O model does not require post-processing to align the label sequences during inference since the output sequence preserves the alignment between the word and corresponding annotation labels; alignment is only needed for the data preparation stage during training to produce the appropriate target sequences. For this reason, we used the O2O model in this study.

This paper proposes to use a state-of-the-art Transformer-based E2E ASR system (Karita et al., 2019) for the O2O model with a single sequence, instead of CTC-based approaches which are frequently supported (Audhkhasi et al., 2018; Ghannay et al., 2018). Compared with the CTC-based systems, this approach can explicitly model the relationship between the output labels thanks to the autoregressive decoder network, similar to the conditional chain rule model in Fig. 1(b). We also demonstrate improved performance compared to the CTC-based systems. Another contribution is that we conducted an extensive empirical evaluation to analyze and demonstrate the utility of our approach. For example, we applied the method to English and Japanese ASR tasks in which phonemic transcripts and POS tags are simultaneously produced. Our approach predicts linguistic annotations correctly even though corresponding graphemes are wrong, while the pipeline approach, in which NLP-based methods are applied to a hypothesized ASR transcript, fails. This feature is helpful for the downstream NLP system like slot filling or intent detection. Besides, our approach is suitable for on-device applications because the E2E model archives small-footprint prediction (Pang et al., 2018). Note that our primary goal is to provide aligned transcripts and linguistic annotations with minimal degradation in ASR performance. We are *not* aiming to improve ASR performance. The features of the proposed method are summarized as follows:

- The proposed Transformer-based O2O model can explicitly model the relationship between the output graphemes and corresponding linguistic annotations, unlike the O2M and CTC-based O2O models.
- Our approach does not require additional alignment post-processing across the transcriptions and the sequence of the linguistic annotations during inference.
- We can easily combine the proposed O2O model with downstream NLP tasks and also conduct an intuitive error analysis (e.g., detecting the error caused due to the homonym by checking the word and the corresponding phoneme output).

2 Existing E2E ASR system

2.1 E2E ASR

The objective of E2E ASR is to estimate the output token sequence $\mathbf{y} = \{y_m \in \mathcal{Y}\}_{m=1}^{L}$ from input feature sequences $\mathbf{X} = \{\mathbf{x}_i \in \Re^{D^{\text{in}}}\}_{i=1}^{I^{\text{in}}}$. Here, D^{in}

and I^{in} denote the number of the dimension of the input feature and the length of the input sequence, respectively; and L and $\mathcal{Y}$ denote output sequence length and the token vocabulary. To predict the output token sequence, an NN is trained to maximize the following conditional likelihood objective function:

$$\begin{aligned}\mathcal{L} &= \log p(\mathbf{y}|\mathbf{X}) \\ &= \sum_{m=1}^{M} \log p(y_m|y_1, \ldots, y_{m-1}, \mathbf{X}). \end{aligned} \quad (1)$$

During run-time, the ASR output $\hat{\mathbf{y}}$ is predicted by

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y}\in\mathcal{Y}^*} \log p(\mathbf{y}|\mathbf{X}), \quad (2)$$

where $\mathcal{Y}^*$ denotes a set of all possible hypotheses.

The Transformer (Vaswani et al., 2017) is a state-of-the-art NN architecture that can be used to maximize Eq. (1). The Transformer consists of two NNs: The Encoder network and the Decoder network. Let I^{emb} and D^{emb} be the sequence length and dimension of the acoustic embedding. The Encoder network generates a sequence of embeddings of the acoustic information $\mathbf{E} = \{\mathbf{e}_i \in \Re^{D^{\text{emb}}}\}_{i=1}^{I^{\text{emb}}}$ from input feature sequences, i.e. $\mathbf{E} = \text{Encoder}(\mathbf{X})$. The Decoder network predicts the output of the M-th step y_M given a sub-sequence, including the current output $\bar{\mathbf{y}} = \{y_1, \cdots, y_{M-1}\}$ and $\mathbf{E}$, i.e. $y_M = \text{Decoder}(\bar{\mathbf{y}}, \mathbf{E})$. This conditional autoregressive modeling function is particularly important in this paper since it can explicitly model the relationship between output labels, unlike CTC.

2.2 E2E ASR to predict two or more sequences

This study aims to estimate a word/morpheme sequence and linguistic annotations, such as phonemic transcripts or POS tags, simultaneously. Here, we define the number of sequences including the subword sequence $\mathbf{y}_1$ in Section 2.1 and additional linguistic annotation sequences as $\mathbf{y}_2, \ldots, \mathbf{y}_K$. To predict both transcriptions and linguistic annotations, the NN is trained to maximize the following log-likelihood of the joint probability:

$$\mathcal{L} = \log p(\mathbf{y}_1, \cdots, \mathbf{y}_K|\mathbf{X}) \ , \quad (3)$$

where $\mathbf{y}_k = \{y_{1,k}, \cdots, y_{M_k,k}|y_{m,k} \in \mathcal{Y}_k\}$ denotes an M_k-length sequence of the k-th type of tokens or linguistic annotations and $\mathcal{Y}_k$ denotes a set of the corresponding tokens or symbols. In the rest of this subsection, we explain the following existing models to maximize Eq. (3): the O2M model trained with multi-task learning and the O2O model trained with the conditional chain mapping.

2.2.1 O2M model trained with multi-task learning

One frequently used NN architecture (Ueno et al., 2018; Gowda et al., 2019; Sanabria and Metze, 2018; Adams et al., 2019) that maximizes Eq. (3) is the O2M model trained with multi-task learning. Fig. 1(a) shows the architecture of the model. The O2M model outputs several types of sequences independently. In other words, multi-task learning is derived by assuming conditional independence of output token types for Eq. (3), as follows:

$$\begin{aligned}\mathcal{L} &= \log \prod_{k=1}^{K} p(\mathbf{y}_k|\cancel{\mathbf{y}_1, \cdots, \mathbf{y}_{k-1}}, \mathbf{X}) \quad (4)\\ &= \sum_{k=1}^{K}\sum_{m=1}^{M_k} \log p(y_{m,k}|\bar{\mathbf{y}}_{1:m-1,k}, \mathbf{X}) \ , \quad (5)\end{aligned}$$

where $\bar{\mathbf{y}}_{1:m-1,k} = \{y_{1,k}, \cdots, y_{m-1,k}\}$ denotes a sub-sequence of the k-th type of tokens or linguistic annotations up to $m-1$. The line crossing part of Eq. (4) represents that the sequences $\mathbf{y}_1, \cdots, \mathbf{y}_{k-1}$ are neglected by assuming conditional independence. The purpose of this study is to predict words/morphemes and aligned linguistic annotation jointly. Since the O2M model deals with different lengths of sequences, post-processing is needed to align the multiple sequences. Also, Eq. (4) shows that multi-task learning assumes conditional independence, but transcripts and linguistic annotations are often conditionally dependent. Hence the O2M model is not ideal for this study.

2.2.2 O2O model trained with conditional chain mapping

The O2O model trained with a conditional chain mapping (Fujita et al., 2020a; Shi et al., 2020) can also be used to maximize Eq. (3). Fig. 1(b) shows the architecture of this model. This model predicts the different sequence types sequentially, each time conditioning on all previously decoded sequence types $1 \ldots k-1$. Different from the multi-task training loss (Eq. (4)) used in the O2M model, the O2O conditional chain mapping model is trained to maximize the joint log-likelihood (Eq. (3)) via a recursive expansion of the probabilistic chain rule. This model *does not* require or assume conditional independence between sequence types. Formally, the O2O model is trained to maximize the following loss function:

$$\begin{aligned}\mathcal{L} &= \log \prod_{k=1}^{K} p(\mathbf{y}_k | \mathbf{y}_1, \cdots, \mathbf{y}_{k-1}, \mathbf{X}) \\ &= \sum_{k=1}^{K} \sum_{m=1}^{M_k} \log p(y_{m,k} | \bar{\mathbf{y}}_{m-1,k}, \bar{\mathbf{Y}}_{1:k-1}, \mathbf{X}) \,, \end{aligned} \tag{6}$$

where $\bar{\mathbf{Y}}_{1:k-1} = \{\mathbf{y}_1, \cdots, \mathbf{y}_{k-1}\}$ denotes $(k-1)$ sequences. While this approach can explicitly model inter-sequence dependencies, it still requires post-processing to align the output sequences.

3 Proposed E2E ASR system

3.1 Framework

Fig. 1(c) depicts the proposed *single* sequence O2O E2E ASR model. The single sequence O2O model predicts the word/morpheme and the corresponding linguistic annotations simultaneously by regarding multiple sequences as a single sequence.

In the single sequence representation, the K output sequences are collapsed into a *single* sequence of S segments. The i-th segment $\mathbf{s}_i$ consists of a fixed order of K jointly aligned sub-sequences. Let $\bar{\mathbf{y}}_{i,k}$ be the i-th sub-sequence of the k-th type of tokens or annotations from index $B(i,k)$ to $E(i,k)$, i.e., $\bar{\mathbf{y}}_{i,k} := \bar{\mathbf{y}}_{B(i,k):E(i,k),k}$. Then, $\mathbf{s}_i = (\bar{\mathbf{y}}_{i,1}, \ldots, \bar{\mathbf{y}}_{i,K})$ denotes the i-th variable-length segment composed of aligned graphemic and linguistic annotation sub-sequences. Equation 8 shows how the K sequences are collapsed into a single sequence of composed of segment $\mathbf{s}_i$.

$$\begin{pmatrix} \mathbf{y}_1 \\ \vdots \\ \mathbf{y}_K \end{pmatrix} = \begin{pmatrix} \bar{\mathbf{y}}_{1,1}, \ldots, \bar{\mathbf{y}}_{S,1} \\ \vdots \\ \bar{\mathbf{y}}_{1,K}, \ldots, \bar{\mathbf{y}}_{S,K} \end{pmatrix} \tag{7}$$

$$= \left(\mathbf{s}_1, \ldots, \mathbf{s}_S\right) . \tag{8}$$

To obtain $\mathbf{s}_i$, we use existing annotation tools or manual annotations to jointly align the *training* sets of the K output sequence types. These segments are used as training targets in an auto-regressive prediction task. In this way, our model implicitly learns to simultaneously predict and align K output sequences from an input $\mathbf{X}$. We discuss further details of the data preparation in Section 3.2.

Letting y_i^* denote elements of the collapsed single-sequence representation $\left(\mathbf{s}_1, \ldots, \mathbf{s}_S\right)$, the joint log-likelihood (Eq. (3)) can be written as

$$\begin{aligned}\mathcal{L} &= \log p(\mathbf{y}_1, \ldots, \mathbf{y}_K | \mathbf{X}) \\ &= \sum_{m=1}^{M^*} \log p(y_m^* | y_1^*, \ldots, y_{m-1}^*, \mathbf{X}) \,. \end{aligned} \tag{9}$$

Note that this form is almost equivalent to the single sequence objective function in Eq. (1) except for the variable y_m^* takes values from the union of the K symbol sets that represent the K output sequences and the length of this sequence $M^* = \sum_{k=1}^{K} M_k$, is the sum of the lengths of the K output sequences.

This framework has various benefits compared with the existing frameworks described in Section 2. Similar to the O2O model trained with the conditional chain mapping in Section 2.2.2, this framework does not assume the conditional independence between output labels and has the flexibility to model the dependency between words/morphemes and linguistic annotations. Related works are using the O2O model, e.g., (Yadav et al., 2020), but they are based on CTC and do not consider such an explicit output dependency. Also, the proposed method using Transformer can preserve a relationship between the word/morpheme and the corresponding linguistic annotations across the sequence based on the aligned representation $\mathbf{s}_i$ in Eq. (8). Finally, this framework is equivalent to the original single-sequence objective function, and we can use an existing strong sequence-to-sequence model (transformer in this paper) without any modifications of the algorithm. The only process is to prepare the collapsed single sequence composed of $\mathbf{s}_i$, which is discussed in the next section.

3.2 Data preparation

This section describes how we prepare the collapsed single sequence composed of $\mathbf{s}_i$ in Eq. (8). We explain this data preparation with both English (TED-LIUM release 2 (TEDLIUM2) (Rousseau et al., 2014)) and Japanese (corpus of spontaneous Japanese (CSJ) (Maekawa et al., 2000)) data as an example. The sequence type includes the graphemic and phonemic transcripts[1], as well as the POS tags.

Fig. 2 shows how to obtain the target sequence. First, we predict sequences of phonemes and POS tags from the graphemic sequences using manually

[1] In the Japanese task, we used the kana character, a syllabic character, and this paper regards it as a phoneme.

(a)	Graphemes	: I // go // home
(b)	Phonemes	: AY // G OW // HH OW M
	POS tags	: Noun // Verb // Noun
(c)	Phonemes	: <Ph12> // <Ph21> <Ph31> // <Ph16> <Ph31> <Ph34>
	POS tags	: <Pos3>// <Pos5> // <Pos3>
(d)	Graphemes ($\mathbf{y}_1$)	: I $\bar{\mathbf{y}}_{1,1} = (y_{1,1})$ go $\bar{\mathbf{y}}_{2,1} = (y_{2,1})$ home $\bar{\mathbf{y}}_{3,1} = (y_{3,1})$
	Phonemes ($\mathbf{y}_2$)	: $\underbrace{\text{<Ph12>}}_{\bar{\mathbf{y}}_{1,2} = (y_{1,2})}$ $\underbrace{\text{<Ph21> <Ph31>}}_{\bar{\mathbf{y}}_{2,2} = (y_{2,2}, y_{3,2})}$ $\underbrace{\text{<Ph16> <Ph31> <Ph34>}}_{\bar{\mathbf{y}}_{3,2} = (y_{4,2}, y_{5,2}, y_{6,2})}$
	POS tags ($\mathbf{y}_3$)	: <Pos3> $\bar{\mathbf{y}}_{1,3} = (y_{1,3})$ <Pos5> $\bar{\mathbf{y}}_{2,3} = (y_{2,3})$ <Pos3> $\bar{\mathbf{y}}_{3,3} = (y_{3,3})$
(e)		$\underbrace{\text{I <Ph12> <Pos3>}}_{\mathbf{s}_1 = (\bar{\mathbf{y}}_{1,1}, \bar{\mathbf{y}}_{1,2}, \bar{\mathbf{y}}_{1,3})}$ $\underbrace{\text{go <Ph21> <Ph31> <Pos5>}}_{\mathbf{s}_2 = (\bar{\mathbf{y}}_{2,1}, \bar{\mathbf{y}}_{2,2}, \bar{\mathbf{y}}_{2,3})}$ $\underbrace{\text{home <Ph16> <Ph31> <Ph34> <Pos3>}}_{\mathbf{s}_3 = (\bar{\mathbf{y}}_{3,1}, \bar{\mathbf{y}}_{3,2}, \bar{\mathbf{y}}_{3,3})}$
(f)		I <Ph12><Pos3> go <Ph21><Ph31> <Pos5> h ome <Ph16> <Ph31> <Ph34><Pos3>

Figure 2: Example of the target sequence for the proposed model. <Ph·>, <Pos·>, and // denote symbols of phonemes and graphemes, and the word boundary, respectively. (a) the graphemic sequence; (b) sequences of phonemes and POS tags; (c) sequences whose annotations are mapped into specific symbols; (d) sub-sequences of graphemes, phonemes, and POS tags; (e) target sequence; (f) target sequence applied byte-pair encoding (Kudo and Richardson, 2018).

annotated labels or annotation tools (Fig. 2(a),(b)). For the Japanese data, we use the annotation labels provided in the corpus. Note that some of the POS tags are estimated using a morphological analysis model. For the English data, we obtain these sequences from the pronunciation dictionary provided in the corpus and WordNet (Miller, 1998), respectively. Some words in the vocabulary have two or more pronunciations in the pronunciation dictionary. To obtain phoneme sequences, we randomly selected a single pronunciation per word from the candidate pronunciations. Since in WordNet, 57 % of the words in the corpus are not annotated with the POS tags, we annotated these labels with the output of the POS tagging system (Loper and Bird, 2002). Next, we replaced these phonemes and POS tags with special symbols (Fig. 2(c)) to distinguish them from the grapheme symbols. Third, we split graphemic and linguistic annotation sequences at word boundaries and obtain sub-sequences ($\bar{\mathbf{y}}_{i,k}$ in Eq. (8)) (Fig. 2(d)). Then sub-sequences are aggregated with the segments ($\mathbf{s}_i$ in Eq. (8)) and collapsed into the target sequence in the manner of Eq. (8) (Fig. 2(e)). For the English data, we applied byte-pair encoding (BPE) (Kudo and Richardson, 2018) to the collapsed target sequence (Fig. 2(f)).

4 Experiments

4.1 Experimental setup

4.1.1 E2E ASR

We built a Transformer-based ASR system using the ESPnet toolkit (Watanabe et al., 2018). The Transformer architecture and hyper-parameters for training/decoding are based on existing recipes in ESPnet. We investigated three models: self-attention-based CTC (Pham et al., 2019), the Transformer (Dong et al., 2018), and a hybrid Transformer trained with an auxiliary CTC objective (Transformer+CTC) (Karita et al., 2019). The CTC model was used in prior studies based on O2O models, e.g., (Audhkhasi et al., 2018; Yadav et al., 2020). During training, the CTC model was regularized with the Transformer decoder in the multi-task learning fashion similar to Transformer+CTC. Such regularization techniques yield a significant improvement over a pure CTC baseline (Fujita et al., 2020b).

For the training of Transformer+CTC, we applied joint CTC training to improve performance (Karita et al., 2019). For CTC-based decoding, we used the greedy search algorithm. For Transformer decoding, we used the beam search algorithm and tuned search parameters using the development set. For the Transformer+CTC model, we applied Transformer/CTC joint decoding (Karita et al., 2019). and tuned the weights of the objective using the development set. Note that the language model shallow fusion (Hori et al., 2018) is not applied since we could not find effectiveness in our preliminary experiment.

4.1.2 Evaluation criteria

We evaluate the performance of the proposed method using the character error rate (CER), phoneme error rate (PER), and word error rate

(WER). CER and WER measure the quality of graphemic transcripts in Japanese and English respectively. PER is used to evaluate the quality of phonemic transcripts in both languages. This study aims to incorporate linguistic annotation prediction into the state-of-the-art Transformer-based E2E ASR. We computed the CER/WER/PER to verify that the E2E model can perform ASR adequately even though the additional downstream NLP tasks are incorporated.

To obtain a sequence with alignment ($\mathbf{s}_i$ in Eq. (8)) on the inference stage, grapheme, phoneme, and POS should be generated in the same order as the training stage. To confirm this, we define annotation structure accuracy (ASA) as a metric. We can compute the correct number of the predicted structure and compute the accuracy. For example, the correct order of the output must follow the following grapheme-phoneme-POS order:
<s> I <Ph12> <Pos3> go <Ph21> <Pos5> </s>
where <s> and </s> denote the start and end symbols of a sentence, respectively. However, our sequence-to-sequence model does not have such explicit output constraints and it possibly outputs the following wrong order of the sequence:
<s> I <Ph12> <Pos3> go <Pos5> </s>
Thus, the second case has 5 correct transition counts among 6 total transition counts, and we can compute the accuracy as 5/6. We assume the transition from "go" to <Pos5> is incorrect.

To evaluate Japanese ASR's word segmentation performance, we measure the precision p, recall r, and F-value f of the hypothesized segmentation compared to the ground-truth segmentation. Let N^{hyp}, N^{ref}, and N^{cor} be the numbers of the predicted graphemes, the graphemes of the reference, and the graphemes whose predicted linguistic annotation is correct, respectively. The precision p, recall r, and F-value f are defined as follows: $p = N^{\mathrm{cor}}/N^{\mathrm{hyp}}$, $r = N^{\mathrm{cor}}/N^{\mathrm{ref}}$, $f = 2pr/(p+r)$. We only compared 1,919 utterances whose reference and hypothesis transcripts are exactly matched in order to ignore the effect of the ASR errors.

Additionally, hypothesized ASR transcripts and reference transcripts are aligned with graphemes, and we computed an annotation accuracy to measure the performance of the linguistic annotation. Let N^{in} and N^{cor} be the number of input words whose estimated grapheme is correct and the words whose estimated grapheme and linguistic annotations are correct, respectively. The accuracy is computed by $N^{\mathrm{cor}}/N^{\mathrm{in}}$. Since we do not deal with the words whose grapheme is predicted incorrectly by ASR for computing the annotation accuracy, the annotation accuracy is robust to the ASR error.

Since the above measures for the word segmentation and linguistic annotation do not consider the ASR errors, we finally computed the following measures using all of the utterances (i.e., including ASR errors): normalized edit distance, precision, recall, and F-values.

4.1.3 Baseline of the linguistic annotation

To compare the linguistic annotation performance, we prepared a pipeline system, i.e., ASR followed by an NLP-based linguistic annotation. In the pipeline system, the separated model of CTC+Transformer first predicts graphemic sequences. Then, the linear SVM with L2 normalization, trained using KyTea (Graham and Mori, 2010), predicts word boundaries and linguistic annotation from the predicted sequences. To train KyTea, we only used the transcriptions in the ASR training set to perform a fair comparison to the proposed method.

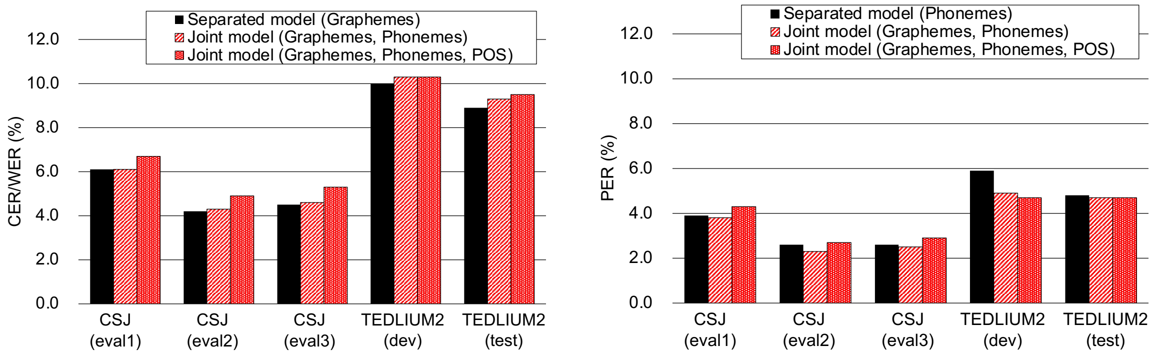
The pipeline system for the Japanese task requires word segmentation before predicting linguistic annotations. The proposed ASR, on the other hand, achieves word segmentation and linguistic annotations simultaneously. Additionally, the proposed ASR achieves these estimates using graphemic information and acoustic information, but the pipeline system uses only the graphemic information. Hence, we expect that the proposed method can predict better word boundary and linguistic annotations for the sentence, which is hard to estimate only from graphemic information. Besides, our model might predict linguistic annotations correctly even though its transcripts are mispredicted, while the pipeline approach fails to predict linguistic annotation when the hypothesized ASR transcriptions include ASR errors. It is helpful for the downstream NLP-based system like slot filling or intent detection.

4.2 Performance of speech recognition

We evaluated ASR performance to confirm the proposed method can produce high-quality transcriptions and linguistic annotations. Note that our primary goal is to simultaneously predict transcription and linguistic annotations by keeping sufficient performance, not improving the ASR performance it-

		CSJ						TEDLIUM2			
		eval1		eval2		eval3		dev		test	
outputs	**model**	**CER**	**PER**	**CER**	**PER**	**CER**	**PER**	**WER**	**PER**	**WER**	**PER**
graphemes, phonemes	CTC (baseline)	7.4	5.0	5.5	3.1	5.9	3.3	15.7	7.3	15.6	7.7
	Transformer	6.9	4.4	4.7	2.6	6.1	3.7	15.8	9.3	15.0	9.1
	Transformer+CTC	**6.1**	**3.8**	**4.3**	**2.3**	**4.6**	**2.5**	**10.3**	**4.9**	**9.3**	**4.7**
graphemes, phonemes,POS	CTC (baseline)	10.0	7.0	7.3	4.4	8.3	5.1	15.8	7.2	14.9	7.0
	Transformer	**6.4**	**4.1**	**4.7**	**2.7**	**5.2**	3.0	14.6	8.8	13.5	8.2
	Transformer+CTC	6.7	4.3	4.9	**2.7**	5.3	**2.9**	**10.3**	**4.7**	**9.5**	**4.7**

Table 1: Comparison between CTC models, Transformer models, and Transformer+CTC models. **CER**, **PER** and **WER** denote character error rate, phoneme error rate, and word error rate, respectively. The CTC model was extensively used in prior work (e.g., Ghannay et al., 2018). Transformer+CTC refers to the Transformer with joint CTC training and decoding (Watanabe et al., 2018).



(a) CER/WER

(b) PER

Figure 3: Comparison between separated model and proposed joint modeling. The separated model predicts graphemes or phonemes; the joint model predicts graphemes and linguistic annotations simultaneously.

self. Table 1 and Fig. 3 show the ASR performance of the Japanese (CSJ) and English (TEDLIUM2) tasks.

First, we discuss which model architecture is appropriate for predicting the grapheme and phoneme sequences. Table 1 shows the Transformer or Transfromer+CTC achieves better performance compared to the CTC model, which corresponds to the conventional method. This means that the Transformer is better for predicting transcriptions and linguistic annotations (phoneme in this experiment) than CTC thanks to the explicit dependency modeling, as discussed in Section 3. Since Transformer+CTC yields better or equivalent performance than the Transformer, we used Transformer+CTC architecture as a base model in the rest of this paper (refer to as a joint model).

Second, we discuss whether the proposed joint models predict the grapheme and phoneme with sufficient performance. To confirm that, we trained two separate models, which predict either a grapheme sequence or a phoneme sequence. Since Transformer+CTC yields better performance than the CTC model and Transformer, we used Transformer+CTC architecture as a base model. Fig. 3 shows that the proposed joint model is almost comparable to the separated model, especially when it predicts both graphemes and phonemes. When the joint model prediction includes the POS tag, we observed a slight degradation, especially in the Japanese task. However, such degradation is still less than 1%, and we can conclude the proposed O2O model of Transformer+CTC can predict graphemes and phonemes simultaneously with sufficient performance. We would emphasize that the proposed joint model can have alignment between grapheme/phoneme/POS while the conventional separated model can not.

4.3 Performance of the annotation structure prediction

As we discussed in Section 4.1.2, we computed the annotation structure accuracy (ASA), and it turns out that its range was from 98.9 % to 100.0 %. This means that the proposed joint model can consistently predict transcriptions and the linguistic annotations in the correct order almost perfectly. We found that almost all errors of the transition occurred in the last word, which might be caused by beam search errors.

Reference:	ピッチ/piqchi/noun	と/to/particle	**スペクトラ**/supekutora/repetition	**スペクトル**/supekutoru/noun	包絡/ho:raku/noun	⋯
Pipeline :	ピッチ/piqchi/noun	と/to/particle	**スペクトラスペクトル**/UNK/UNK		包絡/ho:raku/noun	⋯
Proposed :	ピッチ/piqchi/noun	と/to/particle	**スペクトラ**/supekutora/repetition	**スペクトル**/supekutoru/noun	包絡/ho:raku/noun	⋯

(a) Transcription including repetition.

Reference:	その/sono/adnominal	後/**go**/noun	音楽/oNgaku/noun	番組/baNgumi/noun	が/ga/noun	全盛/zeNse:/noun	⋯
Pipeline :	その/sono/adnominal	後/**ato**/noun	音楽/oNgaku/noun	番組/baNgumi/noun	が/ga/noun	全盛/zeNse:/noun	⋯
Proposed :	その/sono/adnominal	後/**go**/noun	音楽/oNgaku/noun	番組/baNgumi/noun	が/ga/noun	全盛/zeNse:/noun	⋯

(b) Transcription including heteronym.

Figure 4: Examples of the estimated transcription. X, in X/Y/Z, denotes graphemes, and Y, Z denotes phonemes and POS tags, respectively.

System	Precision	Recall	F-value
Pipeline	99.6	99.5	99.6
Proposed	**99.8**	**99.8**	**99.8**

Table 2: Word segmentation performance in CSJ (%). **Pipeline** is ASR predicting transcriptions followed by the NLP-based linguistic annotation system (Graham and Mori, 2010). **Proposed** predicts graphemes and phonemes followed by POS tags from speech. Note that we used only the sentences whose hypothesized ASR transcript is predicted correctly for evaluation.

	CSJ		TEDLIUM2	
System	**Phoneme**	**POS**	**Phoneme**	**POS**
Pipeline	99.4	99.0	99.8	99.7
Proposed	**99.8**	**99.4**	**99.9**	**99.9**

Table 3: Accuracy of predicting linguistic annotation (%). Note that we removed the graphemes which do not appear in the reference from the evaluation to ignore the effect of the ASR errors.

4.4 Performance of word segmentation and predicting linguistic annotations

We evaluated the performance of word segmentation and linguistic annotations using the output of the proposed ASR, which predicts graphemes, phonemes, and POS tags. Note that we did not compute the word segmentation performance in the English task because the English sentences include word boundaries.

Tables 2 and 3 show the performance of the word segmentation and of the predicting linguistic annotations, respectively. Note that these results *do not* consider the ASR error. These tables show that the proposed ASR system achieves better word segmentation and predicts linguistic annotations better than the pipeline system. To compute accuracy, we used 41k and 65k morphemes for CSJ and TEDLIUM2, respectively, and we consider the number of samples is enough to show our model is better than the pipeline approach.

	CSJ		TEDLIUM2	
System	**Phoneme**	**POS**	**Phoneme**	**POS**
Pipeline	**0.08**	0.06	**0.10**	0.08
Proposed	**0.08**	**0.05**	**0.10**	**0.07**

Table 4: Normalized edit distance averaged over the whole evaluation set. Note that we consider the effect of the ASR errors.

Table 4 and 5 show the performance which considers ASR errors. Table 4 shows that the proposed ASR system achieves better prediction of the POS tags than the pipeline, even though the proposed system sometimes failed to predict the transcriptions[2]. Table 5 also confirms that the proposed ASR system predicts better performance in the Japanese task. Although the performance of the pipeline system and the proposed ASR system is comparable in the English task, we would like to emphasize that the proposed ASR does not require extra memory for the additional downstream NLP task. This is useful for developing a small footprint system.

Fig. 4 shows some examples that the proposed ASR can estimate the word boundary and phonemes correctly. For example, the first sentence correctly segments the word boundary based on the "repetition" POS tag estimated from the acoustic information. Similarly, the second sentence appropriately chooses the correct pronunciation from the acoustic information.

5 Discussion

5.1 Building pronunciation dictionary

Since our system of E2E ASR can estimate pairs of graphemes and phonemes for each word, we can build a pronunciation dictionary by considering both graphemes to phoneme sequences and acoustic information.

Table 6 shows the entries of the pronunciation dictionary extracted from the output of our system. The first row of the table lists the entries

[2] We conducted Welch's t-test and found a significant difference between the POS values of the pipeline system and the proposed ASR system ($p < 0.01$).

	CSJ			TEDLIUM2		
System	**Precision**	**Recall**	**F-value**	**Precision**	**Recall**	**F-value**
Pipeline	79.4	82.6	81.4	**84.5**	**71.5**	**77.5**
Proposed	**85.0**	**85.6**	**85.3**	82.6	71.2	76.5

(a) Phoneme

	CSJ			TEDLIUM2		
System	**Precision**	**Recall**	**F-value**	**Precision**	**Recall**	**F-value**
Pipeline	79.1	82.6	80.8	**84.5**	**58.3**	**69.0**
Proposed	**84.9**	**84.8**	**84.8**	83.2	**58.3**	68.6

(b) POS

Table 5: Precision, recall, and F-values of the linguistic annotation prediction. We used all of the sentences, including the hypothesized ASR transcript including the ASR error, for evaluation.

		Entries		
(a)	**REF**	kindergarteners/K,IH,N,D,ER,G,AA,R,T,AH,N,ER,Z	overcooked/OW,V,ER,K,UH,K,T	jovial/JH,OW,V,IY,AH,L
	HYP	kindergarteners/K,IH,N,D,ER,G,AA,R,T,AH,N,ER,Z	overcooked/OW,V,ER,K,UH,K,T	jovial/JH,OW,V,IY,AH,L
(b)	**REF**	Himalaya/HH,IH,M,AH,L,**AY**,AH	forest/F,AO,R,**IH**,S,T	object/**AA**,B,JH,EH,K,T
	HYP	Himalaya/HH,IH,M,AH,L,**EY**,AH	forest/F,AO,R,**AH**,S,T	object/**AH**,B,JH,EH,K,T

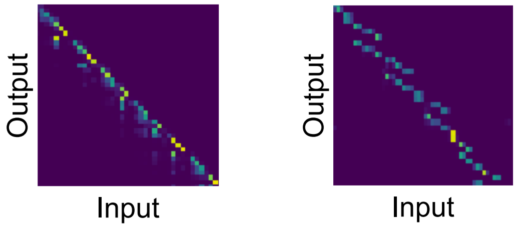
Table 6: Entries of the pronunciation dictionary generated using the output of the proposed E2E ASR. X and Y, in X/Y, denote transcription and phoneme sequence, respectively; and **REF** and **HYP** denote reference and hypothesis, respectively. The row of (a) and (b) list the entries of the out-of-vocabulary word and the entries of the heteronym, respectively.

whose words did not appear in the text of the training set and whose phoneme sequence is estimated correctly. These entries indicate that our system can predict the phonemes of OOV words. The second row of the table shows the entries whose phonemes are different from the reference but exist in the CMU pronunciation dictionary (CMU). In other words, these entries have variations of the phoneme sequence for each word, and the phoneme sequences are predicted correctly. In this study, we removed the phoneme sequence variations for each grapheme from the training set. If the Transformer is trained to predict phoneme sequences using only linguistic information, the phoneme sequences are likely to be mapped into words deterministically. Interestingly, our Transformer recovers the variations of the phoneme sequences for each word. It seems that the acoustic information contributed to predicting the phoneme sequences.

5.2 Attention pattern

One of the Transformer's additional benefits is that we can deduce what is happening inside the Transformer by visualizing the patterns of self-attention and source-target attention weights.

Fig. 5 depicts patterns of the self-attention and source-target attention weights on the third layer of the Decoder network. This figure shows that self-attention changes monotonically but has additional diagonal dotted lines. This means that self-attention uses the multiple (both grapheme and phoneme) output symbols but mostly preserving the order of the sequence. Similarly, it also shows that source-target attention focuses on the acoustic feature of the same time step twice. It shows that both graphemes and phonemes are predicted using the same acoustic features at the same time step, respectively.



(a) Self attention (b) Source-target attention

Figure 5: Example of the attention pattern of the third layer of the Decoder network. The transformer estimates graphemes and phonemes simultaneously.

6 Conclusion and future work

We proposed a novel E2E ASR Transformer system for simultaneously estimating transcriptions and linguistic annotations such as phonemic transcripts or POS tags. This paper showed that the proposed ASR could estimate these features with sufficient performance and also showed reasonable phoneme and grapheme analyses and attention patterns thanks to the aligned output of both output symbols. In future work, we will extend the proposed approach to predict other linguistic annotations such as named entities.

References

The carnegie mellon pronouncing dictionary. [Online]. Available: "http://www.speech.cs.cmu.edu/cgi-bin/cmudict [accessed 21-Aug-2020].

O. Adams, M. Wiesner, S. Watanabe, and D. Yarowsky. 2019. Massively multilingual adversarial speech recognition. In *Proc. ACL*, pages 96–108.

K. Audhkhasi, B. Kingsbury, B. Ramabhadran, G. Saon, and M. Picheny. 2018. Building competitive direct acoustics-to-word models for english conversational speech recognition. In *Proc. ICASSP*, pages 4759–4763.

W. Chan, N. Jaitly, Q. V. Le, and O. Vinyals. 2016. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In *Proc. ICASSP*, pages 4960–4964.

J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio. 2015. Attention-based models for speech recognition. In *NIPS*, volume 28.

L. Dong, S. Xu, and B. Xu. 2018. Speech-Transformer: A no-recurrence sequence-to-sequence model for speech recognition. In *Proc. ICASSP*, pages 5884–5888.

Y. Fujita, S. Watanabe, S. Horiguchi, Y. Xue, J. Shi, and K. Nagamatsu. 2020a. Neural speaker diarization with speaker-wise chain rule. *arXiv:2006.01796*.

Y. Fujita, S. Watanabe, M. Omachi, and X. Chang. 2020b. Insertion-Based Modeling for End-to-End Automatic Speech Recognition. In *Proc. Interspeech*, pages 3660–3664.

S. Ghannay, A. Caubrière,, Y. Estève, N. Camelin, E. Simonnet, A. Laurent, and E. Morin. 2018. End-to-end named entity and semantic concept extraction from speech. In *Proc. SLT*, pages 692–699.

D. Gowda, A. Garg, K. Kim, M. Kumar, and C. Kim. 2019. Multi-task multi-resolution char-to-BPE cross-attention decoder for end-to-end speech recognition. In *Proc. Interspeech*, pages 2783–2787.

N. Graham and S. Mori. 2010. Word-based partial annotation for efficient corpus construction. In *Proc. LREC*, pages 2723–2727.

A. Graves. 2012. Sequence transduction with recurrent neural networks. *arXiv:1211.3711*.

A. Graves and N. Jaitly. 2014. Towards end-to-end speech recognition with recurrent neural networks. In *Proc. ICML*, volume II, pages 1764–1772.

T. Hori, J. Cho, and S. Watanabe. 2018. End-to-end speech recognition with word-based RNN language models. In *Proc. SLT*, pages 389–396.

D. Jurafsky and J. Martin. 2008. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, volume 2.

S. Karita, N. E. Y. Soplin, S. Watanabe, M. Delcroix, A. Ogawa, and T. Nakatani. 2019. Improving Transformer-based end-to-end speech recognition with connectionist temporal classification and language model integration. In *Proc. Interspeech*, pages 1408–1412.

Y. Kubo and M. Bacchiani. 2020. Joint phoneme-grapheme model for end-to-end speech recognition. In *Proc. ICASSP*, pages 6119–6123.

T. Kudo and J. Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proc EMNLP*, pages 66–71.

W. Liang, Y. Tian, C. Chen, and Z. Yu. 2020. MOSS: End-to-end dialog system framework with modular supervision. 34(05):8327–8335.

E. Loper and S. Bird. 2002. NLTK: the natural language toolkit. *Proc. ETMTNLP*, page 63–70.

K. Maekawa, H. Koiso, S. Furui, and H. Isahara. 2000. Spontaneous speech corpus of Japanese. In *Proc. of LREC*, pages 946–952.

G. A Miller. 1998. *WordNet: An electronic lexical database*. MIT press.

R. Pang, T. Sainath, R. Prabhavalkar, S. Gupta, Y. Wu, S. Zhang, and C.-C. Chiu. 2018. Compression of end-to-end models. In *Proc. Interspeech*, pages 27–31.

N.-Q. Pham, T.-S. Nguyen, J. Niehues, M. Müller, and A. Waibel. 2019. Very deep self-attention networks for end-to-end speech recognition. *Proc. Interspeech*, pages 66–70.

A. Rousseau, P. Deléglise, and Y. Estève. 2014. Enhancing the TED-LIUM corpus with selected data for language modeling and more TED talks. In *Proc. LREC*, pages 3935–3939.

R. Sanabria and F. Metze. 2018. Hierarchical multitask learning with CTC. In *Proc. SLT*, pages 485–490.

Laurent El Shafey, Hagen Soltau, and Izhak Shafran. 2019. Joint Speech Recognition and Speaker Diarization via Sequence Transduction. In *Proc. Interspeech*, pages 396–400.

J. Shi, X. Chang, P. Guo, S. Watanabe, Y. Fujita, J. Xu, B. Xu, and L. Xie. 2020. Sequence to multi-sequence learning via conditional chain mapping for mixture signals. In *Proc. NIPS*, volume 33, pages 3735–3747.

E. Simonnet, S. Ghannay, N. Camelin, Y. Estève, and R. D. Mori. 2017. ASR error management for improving spoken language understanding. In *Proc. Interspeech*, pages 3329–3333.

S. Ueno, H. Inaguma, M. Mimura, and T. Kawahara. 2018. Acoustic-to-word attention-based model complemented with character-level CTC-based model. In *Proc. ICASSP*, pages 5804–5808.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.-N. Gomez, L. Kaiser, and I. Polosukhin. 2017. Attention is all you need. In *NIPS*, pages 5998–6008.

S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N. Enrique Yalta Soplin, J. Heymann, M. Wiesner, N. Chen, A. Renduchintala, and T. Ochiai. 2018. ESPnet: End-to-end speech processing toolkit. In *Proc. Interspeech*, pages 2207–2211.

H. Yadav, S. Ghosh, Y. Yu, and R. R. Shah. 2020. End-to-End Named Entity Recognition from English Speech. In *Proc. Interspeech*, pages 4268–4272.