K-Wav2vec 2.0: Automatic Speech Recognition based on Joint Decoding of Graphemes and Syllables

Jounghee Kim, Pilsung Kang*

School of Industrial Management Engineering, Korea University, Seoul, Korea
jounghee.kim@korea.ac.kr, pilsung.kang@korea.ac.kr

Abstract

Wav2vec 2.0 is an end-to-end framework of self-supervised learning for speech representation that is successful in automatic speech recognition (ASR), but most of the work has been developed with a single language: English. Therefore, it is unclear whether the self-supervised framework is effective in recognizing other languages with different writing systems, such as Korean. In this paper, we present K-Wav2Vec 2.0, which is a modified version of Wav2vec 2.0 designed for Korean ASR by exploring and optimizing various factors of the original Wav2vec 2.0. In fine-tuning, we propose a multi-task hierarchical architecture to reflect the Korean writing structure. Moreover, a joint decoder is applied to alleviate the out-of-vocabulary problem. In pre-training, we attempted the cross-lingual transfer of the pre-trained model by further pre-training the English Wav2vec 2.0 on a Korean dataset, considering limited resources. Our experimental results demonstrate that the proposed method efficiently yields robust and better performance on both Korean ASR datasets.

Index Terms: speech recognition, multi-task learning, joint decoding, cross-lingual transfer

1. Introduction

In recent years, self-supervised methodology has shown success in various fields [1, 2]. The Wav2vec 2.0 model [2] is an end-to-end framework of self-supervised learning for automatic speech recognition (ASR), and it has recently been presented as an effective pre-training method to learn speech representations. When followed by fine-tuning with small amounts of labeled data, the Wav2vec 2.0 model has shown remarkable performance in English ASR tasks. However, despite the model’s great success, it is still an open question whether this method can be effective with other languages, because most experiments have been conducted with English datasets such as Librispeech [3] and TIMIT [4]. In this paper, we introduce the way to adapt the Wav2vec 2.0 model to Korean ASR by considering various language-specific features, incorporating an effective fine-tuning architecture and efficient pre-training method.

In the Korean writing system, letters are written in syllabic blocks, and these syllabic blocks are composed of 51 Korean grapheme units. This unique writing system allows us to build a Korean ASR model that is based on either graphemes [5, 6] or syllable blocks [7, 8, 9]. According to previous research that investigated modeling units in Korean ASR tasks, syllable-based models outperform grapheme-based models in most cases [10]. Because grapheme-based models require more combinations of input to predict, they generally underperform. However, syllable-based models also have data sparseness problem for infrequently used syllables and the out-of-vocabulary (OOV) problem when the training data is insufficient. In other languages which suffer from the same issues, previous research [11, 12, 13, 14] has identified methods to overcome these problems using a multi-task learning approach (MTL). Multi-task learning is a method for learning shared representations from different (but related) tasks using different modeling units together. By learning the shared representations between high-level and low-level modeling units, multi-task models alleviate data sparseness issues and achieve better performance [13]. Moreover, some research [14] has introduced recovering methods to substitute OOV words produced from high-level decoding with segments generated in low-level outputs.

Inspired by prior works conducted with other languages, we propose a multi-task hierarchical fine-tuning architecture of Wav2vec 2.0 to reflect the unique relationship that exists in Korean writing between syllables and graphemes. By learning useful intermediate representations, the proposed model can generate multi-level units without sacrificing performance. In the inference step, we used a joint decoding strategy that considers grapheme-level and syllable-level units to find the best sequence from a set of candidates instead of using additional language models. This decoding approach leads to better performance and robust results by alleviating the problem of OOV words in syllable-level outputs in various datasets.

In practice, collecting appropriate unlabeled data for stable training is expensive task. To consider the practical circumstance wherein only limited resources are available, we also experimented with the cross-lingual transfer of the English Wav2vec 2.0 model to Korean ASR tasks. Recently, several works [15, 16] have shown that the cross-lingual transfer methods, pre-training with English and fine-tuning with the other low-resource languages, can be effective in improving the performance of downstream tasks. We adopted the cross-lingual transfer method in ASR to improve the model’s performance with a limited Korean dataset by further pre-training the English Wav2vec 2.0 model. Our further pre-training approach efficiently learns Korean speech representations, taking advantage of learned representations from another language.

2. Backgrounds

In this section, we briefly review the base architecture of Wav2vec 2.0 and its training process for ASR.

2.1. Pre-training Wav2vec 2.0

The base architecture of Wav2vec 2.0 consists of three networks: a feature encoder, a contextual transformer, and a quan-
Figure 1: Overview of our proposed ASR framework.

The feature encoder, composed of a multi-layer convolutional neural network, encodes raw audio $X$ and outputs the latent speech representations $Z$. The contextual transformer, a stack of transformer encoders, learns context representation $C$ by taking latent speech representations as input. The quantization module is used to map latent representations into discretized space $Q$, choosing discrete codebook entries in a fully differentiable way.

In pre-training, a certain portion of latent representations are randomly masked before feeding them into the contextual transformer. The model is trained by solving a contrastive task with masked representations, distinguishing the true quantized latent vector from those discrete latents vectors randomly sampled from other masked time steps. During the pre-training, the model learns contextualized representations only with unlabelled speech audio data.

2.2. Fine-tuning for ASR

For ASR task, a randomly initialized linear layer is added on the top of the pre-trained model. This linear layer takes the contextualized representations of the pre-trained model and generates most probable words. In fine-tuning, connectionist temporal classification (CTC) loss, an approach for sequence labeling without alignment information between output sequences and input audio, is used to train both the linear layer and the pre-trained model.

3. Method

Figure 1 shows the architecture of our Korean ASR system, K-Wav2vec 2.0. The proposed system is a stack of the pre-trained Wav2vec 2.0, the multi-task fine-tuning architecture, and the joint decoder.

3.1. Multi-task hierarchical architecture

The multi-task fine-tuning architecture, which consists of the grapheme encoder and the syllable encoder, is trained by taking contextualized representations learned from the pre-trained Wav2vec 2.0 model as an input. Given raw audio $X$, let $C = c_1, ..., c_F$ be the sequence of encoded audio features, i.e., the contextualized representations of the pre-trained model, where $F \in \mathbb{N}^+$ is the number of encoded audio frames. In the grapheme encoder, a linear layer is adopted to project the encoded features into a grapheme vocabulary $g \in G$, followed by the softmax function to produce the posterior probabilities of grapheme sequences $p(g_j | c_t)$ corresponding to each frame. In the syllable encoder, on the other hand, the encoded features are first fed to a stack of transformer encoders, which converts the encoded features $c_t$ to the sequence of hidden vectors $h_t$ to capture the relation between low-level and high-level information. Then, a linear projection layer and the softmax function are applied to produce the posterior probabilities of syllable sequences $p(s_j | h_t)$, where $s \in S$ is the syllable vocabulary.

Under the conditional independence assumption, the posterior probability of a complete sequence is computed using the syllable outputs or grapheme outputs of each frame:

$$p_{\text{syl}}(l = s_1, ..., s_F | X) = \prod_{f=1}^{F} p(s_f | h_f) \quad (1)$$

$$p_{\text{grap}}(l = g_1, ..., g_F | X) = \prod_{f=1}^{F} p(g_f | c_t) \quad (2)$$

3.2. Multi-task fine-tuning

We used the CTC loss when training the model [17]. The CTC is an approach for sequence labeling, where the lengths of label sequence and output frames are different. Given a ground truth label sequence $Y = y_1, ..., y_t$, the CTC process can expand $Y$ to a set of sequences $\Omega(Y)$ by adding a blank token between consecutive labels and allowing each label to be repeated, where the length of each sequence in the extended set $\Omega(Y)$ and output frames are same. Then the posterior conditional probability of label sequence with the CTC process is computed as follows:

$$p_{\text{syl}}(l | X) = \sum_{l \in \Omega(Y)} p_{\text{syl}}(l | X) \quad (3)$$

The CTC loss is the log posterior conditional probability of label sequence with the CTC process. The objective function of the proposed architecture is the weighted sum of grapheme and syllable-level CTC losses to learn the two tasks simultaneously:

$$L_{\text{MTL}} = \lambda \log p_{\text{syl}}(Y | X) + (1 - \lambda) \log p_{\text{grap}}(Y | X) \quad (4)$$

where $\lambda : 0 \leq \lambda \leq 1$ is a hyper-parameter that controls the trade-off between the importance of syllable and grapheme.

3.3. Joint decoder

In the inference step, the joint decoder combines grapheme-level and syllable-level beam search results to find the most confident sequence within limited practical time. In contrast to prior works [18, 19], we only use outputs of acoustic model which allow us to build ASR system without external language models. To extract candidate sequences from outputs with the CTC process, we use the CTC beam search decoder described in [20]. The beam search decoding iteratively finds candidates over time-steps of CTC output and scores them with given probabilities of each time-step. Given syllable outputs, the beam search decoding returns several most likely sequences, syllable candidates $\hat{S}$, and their corresponding probabilities. The beam search results of grapheme outputs are grapheme candidates $G$ and their probabilities in the same way. The beam search of each level is conducted separately to find their candidates. The objective of the joint decoder is to find the most probable se-
sequence $\hat{Y}$ among them:

$$\hat{Y} = \arg \max_{Y \in \hat{S} \cup \hat{G}} \left\{ \gamma \text{ctc}_\text{syll.}(Y|X) + (1 - \gamma) \text{ctc}_\text{grap.}(Y|X) \right\} \tag{5}$$

where $\gamma : 0 \leq \gamma \leq 1$ is a weight that controls the contribution of results from two different levels to the final output. In the joint decoding process, the proposed model can be more robust than a single encoder (syllable-based decoder or grapheme-based decoder) by adjusting the probability of syllable and grapheme candidate where there is the same candidate in each beam search result $\hat{S} \cap \hat{G}$. The additional candidates to syllable output can be generated by using grapheme candidates, where non-overlapped candidates of grapheme beam search exist $\hat{G} - (\hat{S} \cap \hat{G})$. Because the grapheme modeling unit has less vocabulary but can express more syllables by combining graphemes, the additional candidates from the grapheme beam search result alleviate the OOV problem and the data sparseness problem pertaining to syllable output.

### 3.4. Cross-lingual transfer pre-training

To leverage the ASR performance with limited data, we explore a cross-lingual transfer by further pre-training the existing English model on Korean dataset. The motivation behind this approach is the early successes of further pre-training in natural language processing [21, 22, 23, 24]. They investigated the impact of additional pre-training with various language domains by further pre-training the already pre-trained models on a task-relevant corpus, followed by the classification task of the target domains. Recent research [21] has shown the benefit of further pre-training with the generalized models on target-specific data to specialize models in their target task. As long as speech representations of pre-trained models are known to share generalized features across languages [15], further pre-training on the target language can also benefit by taking advantage of shared information. In the proposed model, we further pre-train the English Wav2vec 2.0, which is the pre-trained on English dataset (960 hours), on Korean dataset (965 hours). When this additional pre-training is completed, the model is fine-tuned with Korean-labeled speech data for the downstream task.

Unlike other well-used cross-lingual pre-training methods in speech domain [15, 16, 25], which use a large amount of multilingual corpus composed of various languages from the initial stage of model pre-training, our proposed two-stage pre-training method utilizes the existing English model. Therefore, the proposed method does not have the disadvantage of putting the target language into the multilingual corpus first and developing it from scratch when the corresponding language is not included in the multilingual corpus.

### 4. Experiment setup

#### 4.1. Datasets

We verify the proposed method with Korean speech datasets, including large-corpus dialog dataset, Ksponspeech [8], and call-based benchmark speech corpus, Clovaccall [9]. The Ksponspeech consists of train, development, evaluation-clean, and evaluation-other, total of 1000 hours. The dual transcription, the phonetic script written with the original sound as possible and the orthographic script written with Korean standard orthographic rules by modifying numeric and abbreviation notation, is provided for downstream tasks in parallel. Following the preprocessing guideline [7], we split the dual transcription into the phonetic and orthographic scripts, and evaluate our method. The Clovaccall has relatively short dialogues, containing 50 hours of label data for training and 1 hour for testing. We randomly sampled 10% of the training set for use as a development set for validation. The amount of Clovaccall training data is relatively small to cover all vocabulary of the evaluation set; there is OOVs and data sparseness problem.

### 5. Results

To investigate the effect of multi-task fine-tuning and decoding strategies, we built six K-Wav2vec 2.0 models as shown in Table 1 and 2. The first column represents the pre-training method with the fine-tuning structure, and the second column denotes the used decoding scheme. The transformer for fine-tuning architecture is identical to the syllable encoder of the multi-task

---

1https://github.com/JoungheeKim/K-wav2vec
Table 1: Evaluation results on the Ksponspeech eval-clean and eval-other sets with dual transcription.

<table>
<thead>
<tr>
<th>Model</th>
<th>Decoding</th>
<th>Eval-clean CER</th>
<th>sWER</th>
<th>Eval-other CER</th>
<th>sWER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ksponspeech - phonetic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Big Transf.</td>
<td>syllable</td>
<td>7.77</td>
<td>12.87</td>
<td>8.26</td>
<td>14.40</td>
</tr>
<tr>
<td>K-Wav2vec 2.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+ Linear</td>
<td>syllable</td>
<td>7.30</td>
<td>12.48</td>
<td>7.81</td>
<td>14.12</td>
</tr>
<tr>
<td>+ Linear</td>
<td>grapheme</td>
<td>6.96</td>
<td>12.01</td>
<td>7.48</td>
<td>13.68</td>
</tr>
<tr>
<td>+ Transf.</td>
<td>syllable</td>
<td>6.98</td>
<td>12.02</td>
<td>7.53</td>
<td>13.84</td>
</tr>
<tr>
<td>+ Multi-task</td>
<td>grapheme</td>
<td>7.29</td>
<td>12.47</td>
<td>7.83</td>
<td>14.20</td>
</tr>
<tr>
<td>+ Multi-task</td>
<td>syllable</td>
<td>6.90</td>
<td>11.87</td>
<td>7.35</td>
<td>13.38</td>
</tr>
<tr>
<td>+ Multi-task</td>
<td>joint</td>
<td>6.88</td>
<td>11.76</td>
<td>7.33</td>
<td>13.27</td>
</tr>
<tr>
<td>S-Wav2vec 2.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+ Transf.</td>
<td>syllable</td>
<td>7.16</td>
<td>12.40</td>
<td>7.64</td>
<td>14.01</td>
</tr>
<tr>
<td>+ Multi-task</td>
<td>joint</td>
<td>7.08</td>
<td>12.16</td>
<td>7.70</td>
<td>14.13</td>
</tr>
<tr>
<td>E-Wav2vec 2.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+ Transf.</td>
<td>syllable</td>
<td>8.58</td>
<td>15.34</td>
<td>9.23</td>
<td>17.61</td>
</tr>
<tr>
<td>+ Multi-task</td>
<td>joint</td>
<td>8.10</td>
<td>14.20</td>
<td>8.81</td>
<td>16.53</td>
</tr>
<tr>
<td>Ksponspeech - orthographic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Big Transf.</td>
<td>syllable</td>
<td>8.46</td>
<td>14.02</td>
<td>9.23</td>
<td>16.31</td>
</tr>
<tr>
<td>K-Wav2vec 2.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+ Linear</td>
<td>grapheme</td>
<td>7.89</td>
<td>13.63</td>
<td>8.90</td>
<td>16.35</td>
</tr>
<tr>
<td>+ Linear</td>
<td>syllable</td>
<td>7.49</td>
<td>13.03</td>
<td>8.43</td>
<td>15.69</td>
</tr>
<tr>
<td>+ Transf.</td>
<td>grapheme</td>
<td>7.48</td>
<td>13.03</td>
<td>8.45</td>
<td>15.67</td>
</tr>
<tr>
<td>+ Multi-task</td>
<td>grapheme</td>
<td>8.05</td>
<td>13.96</td>
<td>9.09</td>
<td>16.75</td>
</tr>
<tr>
<td>+ Multi-task</td>
<td>syllable</td>
<td>7.53</td>
<td>13.09</td>
<td>8.36</td>
<td>15.42</td>
</tr>
<tr>
<td>+ Multi-task</td>
<td>joint</td>
<td>7.54</td>
<td>13.03</td>
<td>8.37</td>
<td>15.38</td>
</tr>
</tbody>
</table>

The second experiment was conducted on the Clovalcall, where a total of OOV syllable occurrence is 41 in the testset. Table 2 shows that various configurations with further pre-trained models significantly outperformed the models introduced in [9] and the baseline model, small transformer, released by [27]. Since our proposed architecture with joint decoding gets a robust performance in both datasets. Considering the various conditions in the real world, the robustness of our proposed architecture can play an important role in a service-level ASR system.

6. Conclusion

In this work, we presented a multi-task fine-tuning architecture with a joint decoder and further pre-training approach for the Korean ASR. Our experiments show that the multi-task model can generate multi-level outputs without performance degeneration, and the joint decoder enhances the ASR performance by overcoming the drawbacks of each modeling unit. Our system achieved the best performance in terms of sWER and the most robust performance in both datasets. We also found the further pre-training approach effective using pre-trained representations of the English model. In the future, we will investigate decoding strategies using acoustic model and language model, which can relieve the OOV problem by additional vocabulary.
7. References


