TranUSR: Phoneme-to-word Transcoder Based Unified Speech Representation Learning for Cross-lingual Speech Recognition

Hongfei Xue¹, Qijie Shao¹, Peikun Chen¹, Pengcheng Guo¹, Lei Xie¹*, Jie Liu²

¹Audio, Speech and Language Processing Group (ASLP@NPU), School of Computer Science, Northwestern Polytechnical University, Xi’an, China
²Huawei Cloud

hxue@mail.nwpu.edu.cn, lxie@nwpu.edu.cn

Abstract

UniSpeech has achieved superior performance in cross-lingual automatic speech recognition (ASR) by explicitly aligning latent representations to phoneme units using multi-task self-supervised learning. This enables the learned representation to transfer well from high-resource to low-resource languages, predicting words directly from these phonetic representations in downstream ASR is challenging. In this paper, we propose TranUSR, a two-stage model comprising a pre-trained UniData2vec and a phoneme-to-word Transcoder. Different from UniSpeech, UniData2vec replaces the quantized discrete representations with continuous and contextual representations from a teacher model for phonetically-aware pre-training. Then, Transcoder learns to translate phonemes to words with the aid of extra texts, enabling direct word generation. Experiments on Common Voice show that UniData2vec reduces PER by 5.3% compared to UniSpeech, while Transcoder yields a 14.4% WER reduction compared to grapheme fine-tuning.

Index Terms: Cross-lingual speech recognition, unified representation learning, phoneme-to-word

1. Introduction

End-to-end ASR systems typically require a large amount of labeled data for training [1], which is difficult to obtain in many languages. This problem makes training of ASR in most low-resource languages a challenging task [2, 3]. Cross-lingual learning is an approach widely used to improve ASR performance in low-resource languages [4, 5, 6], which could learn common pronunciation information from non-target high-resource languages to assist ASR for target low-resource languages. Studies [7, 8, 9, 10] have shown that pre-training on multilingual data, including labeled and unlabeled data from available language, and subsequent fine-tuning on the target low-resource language with labeled data, is an effective cross-lingual learning method.

Most recent research on pre-training for cross-lingual ASR can be classified into two categories. The first category only includes unsupervised loss in the pre-training stage, such as [7, 8, 9, 11, 12, 13]. These methods use much inexpensive, unlabeled multilingual data to learn cross-lingual features. In XLSR [9], a notable representation in the field is presented, which involves pre-training a model using self-supervised loss on 53 languages. The second category involves incorporating a supervised loss during unsupervised pre-training [10, 14, 15]. During the pre-training stage, cross-lingual representations is enhanced using labeled data from high-resource languages. The UniSpeech [10] method combines a supervised CTC loss [16] and a self-supervised contrastive loss [17] to enhance the quality of the learned representations. Similarly, the JUST method [15] integrates a supervised RNN-T loss with two unsupervised losses. For ASR tasks, studies such as UniSpeech [10] and JUST [15] have demonstrated that the above-mentioned second category typically achieves higher performance compared to the first category such as XLSR [9].

The choice of modeling units in supervised loss during the pre-training stage is typically between multilingual grapheme sets and phoneme sets. Some methods, like JUST [15], use grapheme units, while others, like UniSpeech [10], opt phoneme units. To the best of our knowledge, the UniSpeech method is currently state-of-the-art (SOTA) on the cross-lingual Common Voice dataset [18]. During the pre-training stage, the methods using grapheme units face challenges in learning shared cross-lingual representations due to a lack of shared graphemes among different languages. Conversely, using phoneme units could make models more easily learn shared phonetic representations. International Phonetic Alphabet (IPA) [19] is a typical cross-lingual phoneme set that is suited for this situation. During the fine-tuning stage, a phoneme-based model cannot directly generate words, and requires additional fine-tuning with grapheme units. However, it is also challenging to map phonetic representations to grapheme units accurately. In addition, the quantizer in UniSpeech is required to generate phoneme representations, which is difficult because the quantizer lacks sufficient trainable parameters.

This study proposes a Transcoder based Unified Speech Representation (TranUSR) learning for cross-lingual ASR. Our TranUSR is a two-stage approach that includes a pre-trained UniData2vec and a phoneme-to-word (P2W) Transcoder. Firstly, we introduce UniData2vec in UniSpeech [10], which is a Data2vec [20] model. UniData2vec replaces UniSpeech’s quantized discrete representations with continuous and contextual representations from a teacher model, enhancing the self-supervised targets to accurately capture phonetic information. Moreover, the teacher model has the same number of parameters as the student model, resulting in complete targets that are closely related to phonemes. This facilitates the learning of a unified speech representation in supervised learning and self-supervised learning. Secondly, we introduce Transcoder, which is based on the phoneme-to-grapheme model [21, 22, 23]. The Transcoder includes various components, such as a probabilistic vector input component. With these components, the Transcoder can convert encoder’s hypotheses to target language’s words with the help of information at different granules. More importantly, the Transcoder can be trained with large-scale, low-cost text data, making it an attractive solution for low-resource settings. Experiments on the Common Voice dataset show that UniData2vec outperforms the SOTA method.
with an average improvement of relative 5.3% phoneme error rate (PER). Furthermore, Transcoder achieves an average improvement of relative 14.4% word error rate (WER) compared to grapheme fine-tuning.

2. Method

As illustrated in Figure 1 (a), the TranUSR framework comprises two modules: UniData2vec and Transcoder. UniData2vec predicts phoneme probabilities from speech features, and Transcoder for each language converts these probabilities into word-level sentences.

2.1. Problem statement

In our research, we use four types of high and low-resource data, which can be denoted as $L$, $U$, $F$ and $T$. $L = \{(x_i^L, y_i^L), i \in (1, N_L)\}$ is labeled non-target high-resource language data consisting of speech audio and transcript labels. $U = \{(x_j^U), j \in (1, N_U)\}$ is unlabeled target low-resource language data, $F = \{(x_k^F, y_k^F), k \in (1, N_F)\}$ is labeled target language data, and $T = \{(y_l^T), l \in (1, N_T)\}$ is pure text target language data. While $L$, $U$, and $T$ are easy to obtain, $F$ is expensive.

Our task involves four steps: (1) training a UniData2vec base model on $L$, (2) initializing with UniData2vec and training a UniData2vec− model on $U$ to learn cross-lingual phonetically-aware representations, (3) fine-tuning the model on each target low-resource language dataset $F$ to improve phoneme recognition ability, and (4) training a P2W Transcoder model for each language using the large text data $T$.

2.2. UniData2vec

Model structure As shown in Figure 1 (a), our UniData2vec module is composed of a convolutional feature encoder [17], two context Transformer encoders [24], and a linear layer. The convolutional feature encoder is used to map the input speech $X$ to latent representations $Z$. The latent representations $Z$ are then fed into two Transformer encoders: a masked student encoder and an unmasked teacher encoder, which is similar to [20]. The student encoder outputs context representations $C$, while the teacher encoder predicts targets $Y$. Finally, the context representations $C$ are mapped to phoneme units via the linear layer. In our method, the teacher and student encoders have the same amount of parameters, thus their phonetic representations have the same quality, which is more accurate than the quantizer outputs in UniSpeech [10].

Multi-task learning The multi-task learning of unified representation mainly consists of a supervised CTC loss [16] and a self-supervised Smooth L1 (SL1) loss [20]. During this stage, our training object includes three parts: 1) Supervised phoneme recognition on the non-target language, which is achieved by using the CTC loss on $L$ to align the context representations $C$ with phoneme units. 2) Self-supervised representation prediction on the non-target language, which is achieved by using the SL1 loss on $L$ to encourage the student encoder to predict the masked samples’ targets $Y$. 3) Self-supervised representation prediction on the target languages, which is achieved by using the SL1 loss on $U$.

Self-supervised loss The self-supervised SL1 loss encourages the student encoder to predict the teacher’s contextual representations $Y$ of masked samples. Specifically, the teacher encoder uses the exponential moving average (EMA) [20] method to parameterize the model parameters. The training targets $Y$ are constructed based on the output of the top $K$ layers of the teacher encoder, which represent the time steps masked in the student encoder. Given the context representations $C$ and the targets $Y$, we use the SL1 loss to regress these targets:

$$L_s(Y, C) = \begin{cases} \frac{1}{2} \frac{(Y - C)^2}{\beta} & |Y - C| \leq \beta \\ \frac{1}{2} \frac{(Y - C - \frac{1}{\beta})}{\beta} & \text{otherwise} \end{cases},$$

where $\beta$ controls the transition from square loss to L1 loss.

Unified representation Supervised CTC and self-supervised SL1 losses are combined to train a unified representation. During supervised training, the primary loss is CTC, and the context representations $C$ generated by the student encoder correspond to phoneme units. Since the EMA method parameterizes the teacher encoder, the self-supervised targets $Y$ generated by the teacher encoder are also related to phoneme units. Therefore, the representations learned by supervised and self-supervised training can be constrained in the same space without the replacement in UniSpeech [10]. The losses can be
expressed separately as:

\[ L_{\text{UniData2vec}} = \sum_{(x,y) \in L} (L_{\text{ctc}} + \alpha L_s), \]  

\[ L_{\text{UniData2vec}^+} = L_{\text{UniData2vec}} + \sum_{x \in U} L_s, \]  

where \( \alpha \) is a tunable weight of self-supervised loss. In the training of dataset \( \mathcal{L} \), it could be set to \((0, 0.5)\).

**Fine-tuning** During the fine-tuning stage, we use the \( \mathcal{F} \) data to train the CTC loss for phoneme recognition in each target language. This helps the model achieve better phoneme recognition ability on the specific language.

### 2.3. Transcoder

**Model structure** As shown in Figure 1 (b), our Transcoder module consists of a linear layer and multiple Transcoder blocks. Each Transcoder block contains two convolutional layers, two multi-head attention blocks [24], a feed-forward neural network, and multiple residual connections [25], as depicted in Figure 1 (c).

**Probabilistic vector input** P2W models typically use phoneme one-hot vectors as input, which can result in performance degradation when phoneme sequences contain errors. Our Transcoder is trained on correct texts and fine-tuned on UniData2vec’s hypotheses that may contain errors. To accommodate both types of input, we use soft phoneme posterior probabilities instead of hard one-hot vectors as input. When the phoneme sequence is from the correct text, its probability vector and one-hot vector are equivalent. However, the conventional text embedding layer only supports hard one-hot vectors as input. To overcome this limitation, we use a trainable linear layer to learn the text embedding, which enables correct text pre-training and hypothesis fine-tuning.

**Intra-word and inter-word information extraction** Conventional P2W models typically use convolutional layers to extract intra-word phoneme dependencies. However, a single convolutional layer may not be sufficient to handle words of different lengths since they contain varying numbers of phonemes. To address this issue, our Transcoder module uses two convolutional models with different kernel sizes to extract intra-word information at two granularities, thereby improving accuracy for words of varying lengths. Additionally, we use a multi-head attention block to extract inter-word information.

**Alignment constraint loss** The sequence lengths of phonemes, words can differ between languages, and CTC loss is often used to handle inconsistent output and input sequence lengths. However, P2W mapping has clear alignment boundaries, unlike audio-to-word mapping. To further improve the performance of Transcoder, we use cross-entropy (CE) loss to ensure accurate alignment learning. Since CE loss requires inputs and outputs to have the same length, we align the sequences by prefixing words with ‘*’ tags. As an example, for the Italian sentence “vostra casa” we align it as shown in Figure 2. With this alignment, our Transcoder can learn to accurately map phonemes to words.

**Phoneme:** vò s t r a k a z a  ①: vostra  
**Word:** * * * * * ① * * * ②: casa

Figure 2: Sequence alignment using ‘*’ tags as prefixes.
Table 1: PER (%) results on Common Voice for different methods. The L and U datasets are described in detail in Section 3. U_uno: Monolingual unlabelled data. Re-implement: we re-ran the baseline experiment from UniSpeech.

<table>
<thead>
<tr>
<th>Model</th>
<th>pre-train</th>
<th>fine-tune</th>
<th>es</th>
<th>fr</th>
<th>it</th>
<th>ky</th>
<th>nl</th>
<th>ru</th>
<th>tt</th>
<th>avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unlabeled data (U)</td>
<td></td>
<td></td>
<td>168</td>
<td>353</td>
<td>90</td>
<td>17</td>
<td>29</td>
<td>55</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td>UniSpeech [10]</td>
<td>L</td>
<td>1 h</td>
<td>10.9</td>
<td>14.8</td>
<td>15.2</td>
<td>11.4</td>
<td>16.2</td>
<td>16.1</td>
<td>9.6</td>
<td>13.5</td>
</tr>
<tr>
<td>UniSpeech++ [10]</td>
<td>L, U_uno</td>
<td>1 h</td>
<td>5.7</td>
<td>7.9</td>
<td>8.1</td>
<td>6.8</td>
<td>9.3</td>
<td>8.6</td>
<td>6.0</td>
<td>7.5</td>
</tr>
<tr>
<td>UniSpeech (re-implement)</td>
<td>L</td>
<td>1 h</td>
<td>10.4</td>
<td>12.7</td>
<td>13.4</td>
<td>10.2</td>
<td>14.8</td>
<td>15.7</td>
<td>8.8</td>
<td>12.3</td>
</tr>
<tr>
<td>UniData2vec</td>
<td>L</td>
<td>1 h</td>
<td>10.0</td>
<td>13.0</td>
<td>13.3</td>
<td>10.3</td>
<td>15.0</td>
<td>15.1</td>
<td>9.0</td>
<td>12.2</td>
</tr>
<tr>
<td>UniSpeech10</td>
<td>L</td>
<td>10 h</td>
<td>7.2</td>
<td>8.1</td>
<td>10.9</td>
<td>4.3</td>
<td>7.3</td>
<td>7.1</td>
<td>3.3</td>
<td>6.9</td>
</tr>
<tr>
<td>UniData2vec10</td>
<td>L</td>
<td>10 h</td>
<td>7.0</td>
<td>8.2</td>
<td>10.4</td>
<td>4.1</td>
<td>6.2</td>
<td>6.6</td>
<td>3.2</td>
<td>6.5</td>
</tr>
<tr>
<td>UniSpeech++ (re-implement)</td>
<td>L, U_uno</td>
<td>1 h</td>
<td>6.1</td>
<td>7.7</td>
<td>8.5</td>
<td>7.8</td>
<td>10.8</td>
<td>9.8</td>
<td>7.4</td>
<td>8.3</td>
</tr>
<tr>
<td>UniData2vec++</td>
<td>L, U_uno</td>
<td>1 h</td>
<td>5.5</td>
<td>6.5</td>
<td>7.1</td>
<td>6.8</td>
<td>8.0</td>
<td>8.8</td>
<td>7.2</td>
<td>7.1</td>
</tr>
</tbody>
</table>

Table 2: WER (%) comparison of Transcoder and the direct fine-tuning of UniData2vec++. Both the two fine-tuning in the table using 1 hour of data. The BPE model has an output label of 4000 BPE units. The Grapheme model uses grapheme sets from the 400k sentences.

<table>
<thead>
<tr>
<th>Model</th>
<th>it</th>
<th>es</th>
<th>fr</th>
<th>avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>UniData2vec++ + BPE fine-tune</td>
<td>63.5</td>
<td>62.9</td>
<td>69.7</td>
<td>65.4</td>
</tr>
<tr>
<td>UniData2vec++ + Grapheme fine-tune</td>
<td>29.6</td>
<td>29.5</td>
<td>45.3</td>
<td>34.8</td>
</tr>
<tr>
<td>UniData2vec++ + Transcoder</td>
<td>27.4</td>
<td>25.7</td>
<td>36.2</td>
<td>29.8</td>
</tr>
</tbody>
</table>

Figure 3: Quality of cluster assignments measured by Purity and Mutual Information (MI) obtained through k-means clustering on self-supervised targets.

4.2. Transcoder

**P2W transcoder vs. direct fine-tuning** For a model like UniSpeech or UniData2vec that learns phoneme recognition in the pre-training stage, we can transfer it to predict words of target languages by directly replacing the output units and then fine-tuning. In order to demonstrate the advantages of the Transcoder, we compare it with direct fine-tuning. As shown in Table 2, the Transcoder trained with extra text data results in significantly better results than the baseline BPE [30] model, outperforming the grapheme model WER by about 14.4%. Pure text data is cheaper and easier to obtain than paired fine-tuning data, which makes the Transcoder scheme more practical.

Ablation study of Transcoder We conduct ablation experiments to verify the role of each component in the Transcoder. As shown in Table 3, we evaluate the performance of our proposed Transcoder architecture by ablating its key components, including the use of probabilistic inputs, two convolution modules, and CE loss, using correct text and UniData2vec++ hypothesis inputs as evaluation metrics. The results demonstrate that removing each of these components leads to weighted average WER increase of 7.3%, 7.0%, and 63.2%, respectively. The Conformer architecture using CE loss compared to our Transcoder resulted in a higher weighted average WER of 6.5%, demonstrating the effectiveness of our proposed architecture.

Table 3: Ablation study of each component’s impact on WER (%) of the Transcoder. Only Italian text is used in experiment.

<table>
<thead>
<tr>
<th>Input data</th>
<th>Transcoder</th>
<th>Conformer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard</td>
<td>W/o Prob inputs</td>
<td>W/o Two conv</td>
</tr>
<tr>
<td>Correct text</td>
<td>3.2</td>
<td>3.2</td>
</tr>
<tr>
<td>Hypothesis</td>
<td>27.4</td>
<td>29.4</td>
</tr>
</tbody>
</table>

5. Conclusions

This paper presents an effective approach for low-resource languages that utilizes unified representation learning on phonetically-aware representations, followed by a P2W Transcoder. The proposed UniData2vec model achieves an improvement of over 5.3% compared to the SOTA method, while the Transcoder model yields a 14.4% relative reduction in word error rate compared to grapheme fine-tuning. However, UniData2vec++ shows weaker performance for low amounts of unlabeled data. In the future, we will unify unlabeled data from multilingual languages to pre-train.
6. References


