INTERSPEECH 2021
30 August – 3 September, 2021, Brno, Czechia

Momentum Pseudo-Labeling for Semi-Supervised Speech Recognition

Yosuke Higuchi[1,2*], *Niko Moritz*[1], *Jonathan Le Roux*[1], *Takaaki Hori*[1]
[1]Mitsubishi Electric Research Laboratories (MERL), USA
[2]Waseda University, Japan
higuchi@pcl.cs.waseda.ac.jp, {moritz, leroux, thori}@merl.com

Abstract

Pseudo-labeling (PL) has been shown to be effective in semi-supervised automatic speech recognition (ASR), where a base model is self-trained with pseudo-labels generated from unlabeled data. While PL can be further improved by iteratively updating pseudo-labels as the model evolves, most of the previous approaches involve inefficient retraining of the model or intricate control of the label update. We present *momentum pseudo-labeling* (MPL), a simple yet effective strategy for semi-supervised ASR. MPL consists of a pair of *online* and *offline* models that interact and learn from each other, inspired by the mean teacher method. The online model is trained to predict pseudo-labels generated on the fly by the offline model. The offline model maintains a momentum-based moving average of the online model. MPL is performed in a single training process and the interaction between the two models effectively helps them reinforce each other to improve the ASR performance. We apply MPL to an end-to-end ASR model based on the connectionist temporal classification. The experimental results demonstrate that MPL effectively improves over the base model and is scalable to different semi-supervised scenarios with varying amounts of data or domain mismatch.

Index Terms: pseudo-labeling, self-training, semi-supervised learning, end-to-end speech recognition, deep learning

1. Introduction

Advances in deep learning have led to remarkable success in automatic speech recognition (ASR) [1, 2]. Much of the recent progress lies in the end-to-end (E2E) framework [3–5], which directly optimizes speech-to-text conversion and greatly simplifies model training and inference processes. While E2E systems compete with traditional hidden Markov model-based systems [6–8], their performance heavily relies on the availability of a large amount of labeled data (speech-text pairs), which requires great annotation costs and is not always achievable.

In order to compensate for the lack of labeled data, semi-supervised learning has been attracting increasing attention for improving E2E ASR. Semi-supervised learning utilizes labeled data as well as unlabeled data during model training, where the amount of labeled data is in general much smaller than that of unlabeled data. Some early works for semi-supervised E2E ASR focus on training with a reconstruction framework, including approaches based on a text-to-speech model [9–11] or a sequential auto-encoder [12–14]. Others adopted self-supervised pre-training techniques, such as BERT-like mask prediction [15, 16], contrastive loss [17, 18], and feature clustering [19, 20], to promote downstream E2E ASR tasks.

We focus on self-training [21] or *pseudo-labeling* (PL) [22], which has recently been adopted for semi-supervised E2E ASR and shown to be effective [23–32]. In PL, a teacher (base) model is first trained on labeled data and used to generate pseudo-labels for unlabeled data. A student model is then trained using both the labeled and the pseudo-labeled data to obtain better performance than the teacher. To obtain more sophisticated pseudo-labels, external language models (LMs) and beam-search decoding are often incorporated into the labeling process [23, 29]. Data augmentation is also useful for helping the student model with training on the pseudo-labeled data [26–28]. In addition to these extensions of PL, iterative pseudo-labeling (IPL), where the student model (initialized with the teacher model) is continuously trained on iteratively updated pseudo-labels, has shown promising results [25, 26, 30, 31]. In [26], the student model generates pseudo-labels on the fly from unlabeled data in a self-supervised manner. Pseudo-labels are refined as the model learns and the model is improved using the continuously updating pseudo-labels. However, we found this framework unstable when there is a large amount of unlabeled data or a domain mismatch between labeled and unlabeled data, which is likely to be the case in real-world scenarios. While [30] successfully scales IPL to a large amount of data, some heuristic controls are required for updating pseudo-labels.

In this paper, we present a simple, efficient, stable, and scalable semi-supervised learning algorithm for E2E ASR, referred to as *momentum pseudo labeling* (MPL). In MPL, the pseudo-labels are iteratively updated based on an ensemble of models at different time steps within a single training process [33]. MPL consists of *online* and *offline* models that interact and learn from each other, similar to the teacher-student framework in the mean teacher method [34]. The online model is trained to predict pseudo-labels generated on the fly by the offline model. The offline model maintains a momentum-based moving average of the weights of the online model, which can be regarded as an ensemble of the online models at different training steps. Through the interaction between the two models, MPL effectively stabilizes the training with unlabeled data and handles the constant change in pseudo-labels. The advantages of the proposed MPL are summarized as follows:

Simple and efficient: The semi-supervised training is performed in a single stage, where pseudo-labels are generated on the fly and naturally refined without requiring an LM or beam-search decoding. Our approach is easy to implement and we show an effective way for controlling the momentum update which reduces the burden for heuristic tuning.

Stable and scalable: We show the effectiveness of MPL in a variety of semi-supervised scenarios. Our approach is robust to variations in the amount of data and domain mismatch severity, which often pose difficulties in semi-supervised ASR [31].

2. Momentum Pseudo-Labeling

The overall process of the proposed momentum pseudo-labeling is shown in Algorithm 1. We describe MPL in two steps: 1) the supervised training process of a base E2E ASR model, and 2) the proposed semi-supervised training scheme for improving the model using unlabeled data.

*Research conducted during an internship at MERL

http://dx.doi.org/10.21437/Interspeech.2021-571

Algorithm 1 Momentum Pseudo-Labeling

Input:
$\mathcal{D}_{\text{sup}}, \mathcal{D}_{\text{unsup}}$ ▷ labeled and unlabeled data
$\mathcal{A}$ ▷ an ASR model architecture
α ▷ a momentum coefficient

Train a base model P_θ with architecture $\mathcal{A}$ on $\mathcal{D}_{\text{sup}}$ using (2)
Initialize an online model P_ξ and an offline model P_ϕ with P_θ
repeat
  for all $S \in \mathcal{D}_{\text{sup}} \cup \mathcal{D}_{\text{unsup}}$ **do**
    Obtain $X \sim S$
    Obtain $Y = \begin{cases} Y \sim S & (S \in \mathcal{D}_{\text{sup}}) \\ \hat{Y} \sim P_\phi(Y|X) & (S \in \mathcal{D}_{\text{unsup}}) \end{cases}$
    Compute loss $\mathcal{L}$ for $P_\xi(Y|X)$ with (2) or (4)
    Update ξ using $\nabla_\xi \mathcal{L}$
    Update $\phi \leftarrow \alpha\phi + (1-\alpha)\xi$
  end for
until *maximum iterations are reached*
return P_ξ, P_ϕ

2.1. Supervised training of a base model

The objective of E2E ASR is to model a sequence mapping between a T-length input sequence $X = (\mathbf{x}_t \in \mathbb{R}^D | t = 1, \ldots, T)$ and an L-length output sequence $Y = (y_l \in \mathcal{V} | l = 1, \ldots, L)$. Here, $\mathbf{x}_t$ is a D-dimensional acoustic feature at frame t, y_l an output token at position l, and $\mathcal{V}$ a vocabulary. We focus on models based on the connectionist temporal classification (CTC) [3, 35], which has recently been revisited thanks to advances in neural network architectures [36, 37]. CTC predicts a frame-level sequence $Z = (z_t \in \mathcal{V} \cup \{\epsilon\} | t = 1, \ldots, T)$, which is obtained by introducing a special blank token ϵ into the output sequence Y. Based on a conditional independence assumption per frame, CTC models the conditional probability $P(Y|X)$ by marginalizing over all paths (frame alignments) as:

$$P(Y|X) = \sum_{Z \in \mathcal{B}^{-1}(Y)} \prod_{t=1}^{T} P(z_t|X), \tag{1}$$

where $\mathcal{B}^{-1}(Y)$ denotes all possible paths compatible with Y.

Given labeled data $\mathcal{D}_{\text{sup}} = \{(X_n, Y_n) | n = 1, \ldots, N\}$, a base model P_θ with parameters θ is trained with the supervised objective defined by maximum likelihood estimation of (1):

$$\mathcal{L}_{\text{sup}}(\theta) = -\log P_\theta(Y_n | A(X_n)), \tag{2}$$

where $A(\cdot)$ denotes a data augmentation for improving generalization of the model, here SpecAugment [38].

2.2. Semi-supervised training with MPL

The goal of semi-supervised training is to enhance the base model (trained on labeled data $\mathcal{D}_{\text{sup}}$) by making good use of unlabeled data $\mathcal{D}_{\text{unsup}} = \{X_m | m = N+1, \ldots, N+M\}$. MPL achieves this goal through an interaction between *online* and *offline* models. Let us define the online and offline models as P_ξ and P_ϕ, with model parameters ξ and ϕ, respectively. Both models are initialized with the trained base model P_θ.

On unlabeled data $X \in \mathcal{D}_{\text{unsup}}$, the online model is trained using pseudo-labels $\hat{Y}$ generated by the offline model:

$$\hat{Y} = \operatorname*{argmax}_Y P_\phi(Y|X), \tag{3}$$

where argmax is performed based on the best path decoding of CTC [35]. Specifically, the most probable tokens are selected at each frame and an output sequence is obtained by suppressing repeated tokens and removing blank symbols. To generate pseudo-labels with higher quality, beam-search decoding [26] or an LM [29] are often incorporated into (3), but we used neither of the techniques. We observed that beam-search decoding without an LM has little impact on ASR accuracy for CTC-based models. While exploitation of a strong LM plays an important role for pseudo-labeling, we adopt the greedy decoding to keep our approach efficient and avoid over-fitting to LM information as reported in [30, 31]. With unlabeled data $\mathcal{D}_{\text{unsup}}$ and the corresponding pseudo-labels from (3), the objective of the online model is defined in the same manner as (2):

$$\mathcal{L}_{\text{unsup}}(\xi) = -\log P_\xi(\hat{Y}_{N+m} | A(X_{N+m})), \tag{4}$$

where $\mathcal{L}_{\text{unsup}}$ is maximized via a gradient descent optimization. Note that in (4), we apply the data augmentation to an unlabeled input as in [24, 26], aiming for the online model to learn robust prediction of pseudo-labels from the noisy input. In Sec. 3.4, we show that data augmentation is an important factor of MPL.

Assuming labeled data $\mathcal{D}_{\text{sup}}$ is available during the semi-supervised process, the supervised loss $\mathcal{L}_{\text{sup}}(\xi)$ can be incorporated into the training, helping stabilize the online model as it learns from unlabeled data with $\mathcal{L}_{\text{unsup}}(\xi)$. In Sec. 3.5, we demonstrate that MPL is also effective even when trained solely on unlabeled data (i.e., trained only with (4)).

The offline model, on the other hand, accumulates weights of the online model after every update via

$$\phi \leftarrow \alpha\phi + (1-\alpha)\xi, \tag{5}$$

a momentum-based moving average with a momentum coefficient $\alpha \in [0, 1]$. This momentum update makes the offline model evolve more smoothly than the online model. We can thus control the change in pseudo-labels generated on the fly by the offline model at each training step. This is important to prevent pseudo-labels from deviating too quickly from the initial labels generated by the base model and to avoid collapsing to a trivial solution. Indeed, we empirically observe that training is prone to collapse (emitting only blank symbols for unlabeled data) for $\alpha = 0.0$, in which case the online and offline models share parameters and the online model is trained with self-generated pseudo-labels as in [26]. The problem is prominent when there is a domain mismatch between labeled and unlabeled data, as is often the case in real-world deployment. We demonstrate the momentum update's importance in Sec. 3.5.

After training with MPL, both the online and offline models can be used for evaluation, although we use the online model as our default. Their performance is compared in Sec. 3.3.

Tuning the momentum coefficient: Instead of directly tuning α in (5), we design a novel, more intuitive method for deriving an appropriate value of α. Based on (5), the parameters of the offline model after K iterations can be computed as follows:

$$\phi^{(K)} = \alpha^K \phi^{(0)} + (1-\alpha) \sum_{k=1}^{K} \alpha^{K-k} \xi^{(k)}, \tag{6}$$

where $\phi^{(k)}$ and $\xi^{(k)}$ denote the parameters of each model at the k-th iteration, and $\phi^{(0)} = \xi^{(0)} = \theta$. We here assume that it is important to retain some influence of the base model to stabilize the pseudo-label generation. As a measure of this influence, we focus on the term $\alpha^K \phi^{(0)}$ in (6) and define a weight w of the base model in $\phi^{(K)}$ as $w = \alpha^K$, where we consider K as the number of iterations (i.e., batches) in a training epoch. As K can often be in the thousands, small changes in α lead to huge differences in w (e.g., $0.999^{3000} \ll 0.9997^{3000}$), requiring small adjustments on α for different amounts of training data. Instead of directly tuning α for the momentum update, we propose to tune the weight w, which can be regarded as the proportion of the base model retained after a training epoch. Given w and K, α is calculated as $\alpha = e^{(1/K)\log w}$. By controlling the update through w, we expect MPL to be less affected by the amount of training data, which we examine in Sec. 3.5.

Relationship to prior work: Our approach can be considered as a variant of the self-ensembling technique [33] for semi-supervised learning, where a model is trained based on an ensemble of the models at different training steps. Particularly, MPL is inspired by and similar to the mean teacher framework [34], in that model ensembling is performed via a moving average. However, MPL differs from prior work in the following perspectives. 1) MPL uses hard (pseudo-)labels for training with unlabeled data: while soft labels generally contain richer information for promoting a model training, applying a distillation loss to CTC-based ASR systems is known to be problematic [39]; as CTC models emit spiky posterior distributions and predictions are naturally high-confidence, we consider hard labels more suitable for MPL. 2) MPL is a semi-supervised learning framework for E2E ASR: while most prior works focus on classification problems, few have introduced self-ensembling techniques to semi-supervised sequence-to-sequence mapping problems. 3) MPL applies data augmentation (i.e., SpecAugment) to the input only for training the online model, while the offline model generates pseudo-labels in inference mode: since we do not use soft labels in MPL, it is preferable for pseudo-labels to be accurate; moreover, the online model can learn to robustly predict pseudo-labels from noisy input, an effective approach known as consistency training [24, 26, 40].

3. Experiments

3.1. Experimental setup

Data: The experiments were carried out using the LibriSpeech (LS) [41] and TEDLIUM3 (TED3) [42] datasets. LS consists of utterances from read English audio books and contains 960 hours of training data (split into `train-clean-100`, `train-clean-360`, and `train-other-500`). TED3 consists of utterances from English Ted Talks and contains 450 hours of training data (`train-ted3`). We used the standard validation and test sets for each dataset. As input speech features, we extracted 80 mel-scale filterbank coefficients with three-dimensional pitch features using Kaldi [43]. We used 1k word-pieces for tokenizing output texts, which were constructed from the `train-clean-100` texts using SentencePiece [44].

Semi-supervised settings: We used `train-clean-100` for supervised training and treated the other training sets as unlabeled. Based on the base model trained on `train-clean-100` (LS-100), we considered different semi-supervised settings: LS-100/LS-360, an in-domain setting with unlabeled `train-clean-360`; LS-100/LS-860, an in-domain setting with unlabeled `train-{clean-360,other-500}`; and LS-100/TED3, an out-domain setting with unlabeled `train-ted3`.

Training and decoding configurations: All the experiments were conducted using ESPnet [45]. We used the Transformer architecture [46] as E2E ASR model, which consisted of a stack of 12 self-attention layers with the same configurations as in [47][1]. The base model was trained for 150 epochs using the Adam optimizer [48] with $\beta_1 = 0.9$, $\beta_2 = 0.98$, $\epsilon = 10^{-9}$, and Noam learning rate scheduling [46]. We used 25,000 warmup steps and a learning rate factor of 5.0. The MPL training was iterated for 200 epochs and the online model was trained using the Adam optimizer with an initial learning rate of 10^{-3}, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$. For evaluation, a final model was obtained by averaging model parameters over 10 checkpoints with the best validation performance. For decoding with an LM, we trained one long short-term memory (LSTM) layer with 1024 units, using the `train-clean-100` transcriptions combined with the external text data provided by LibriSpeech [41]. The LM was incorporated into decoding via shallow fusion with a beam-size of 20 and an LM weight of 0.5. For decoding without an LM, we performed the best path decoding of CTC [35]. We used $w = 0.5$ for all MPL experiments, leading to $\alpha = 0.99977$ for LS-100/LS360, $\alpha = 0.99989$ for LS-100/LS-860, and $\alpha = 0.99983$ for LS-100/TED3.

[1] Note that the size of our model is relatively small compared to other state-of-the-art works on semi-supervised ASR [25, 30]. Future work will look to confirm the effectiveness of MPL on larger models.

3.2. Main results

Table 1 shows results on the in-domain LS settings in terms of word error rate (WER) and WER recovery rate (WRR) [49]. Topline results are obtained via fully supervised training. Looking at the LS-360 setting results (`A*`), both standard PL [23] and proposed MPL led to a significant improvement over the base model (`L0`). MPL (`A3`) outperformed PL (`A1`) by dynamically updating pseudo-labels using the offline model, instead of fixing them to those generated by the base model [23]. While PL could be improved by training on strong pseudo-labels generated via beam-search decoding with LM (`A2`), MPL achieved comparable performance without the help of LM (`A3`). Moreover, with LM, PL was prone to overfit to the LM training text as reported in [30, 31]; in contrast, MPL successfully increased model generalization, achieving higher WRR of 81.2% on the "other" set. We also investigated an effective way for incorporating an LM into MPL, where PL is first applied to the base model with strong pseudo-labels (as in `A2`) and the pre-trained model is used for better initialization in MPL. With this pre-training method, MPL achieved the highest WRRs of 72.2%/83.0% (`A4`), mitigating the over-fitting to the LM. In the LS-860 setting (`B*`), MPL again outperformed the baseline. While standard PL was not as effective as in the LS-360 setting in terms of WRR (`B1`,`B2` vs. `A1`,`A2`), MPL successfully recovered the same rate of errors (`B3`,`B4` vs. `A3`,`A4`), demonstrating its scalability with respect to the amount of unlabeled data.

Table 2 lists MPL results on the out-domain TED3 setting. Standard PL resulted in a modest improvement with WRR of at most 49.1% (`C1`,`C2`). In contrast, the gain was more substantial for MPL with WRR of 80.8% (`C3`,`C4`), indicating MPL is capable of efficiently adapting the base model to another domain. For PL, due to the domain mistmach, the advantage of utilizing the LM (trained on LS transcriptions) was smaller compared to the in-domain settings (`C2` vs. `A2`,`B2`). MPL, however, succeeded in making use of the LM via the pre-training trick (`C4`).

In all experiments, decoding with an LM further improved performance, with MPL again achieving high WRR.

3.3. Online model vs. offline model

Contrary to previous work [34], we adopted the online model for final evaluation. When we did not use the checkpoint averaging technique [46], the offline model gave better performance than the online model (e.g., 9.4%/22.7% vs. 9.8%/24.0% on dev. sets in the LS-360 setting). Since the offline model is an average of the online models over the training (cf. (5)), the offline model naturally benefited from the model ensembling. However, with checkpoint averaging, both of the models were improved and the performance gap was reduced to almost none (e.g., 9.4%/22.5% vs. 9.4%/22.4% on dev. sets in the LS-360 setting). We used the slightly better online model for evaluation.

3.4. Importance of data augmentation

When applying MPL to the base model (`L0` in Table 1) without SpecAugment, WRRs dropped to 39.7%/48.1% compared

Table 1: *Word error rate (WER) [%] and WER recovery rate (WRR) [%↑] on in-domain LibriSpeech (LS) settings. The results are divided into two sections, depending on whether the LM with beam-search decoding was applied in the final evaluation or not.*

				Decoding without LM						Decoding with LM					
				Dev WER [%]		**Test WER** [%]		**Test WRR** [%↑]		**Dev WER** [%]		**Test WER** [%]		**Test WRR** [%↑]	
Setting	**Method**		**Init.**	clean	other	clean	other	clean	other	clean	other	clean	other	clean	other
LS-100	L0	baseline	–	13.2	31.1	13.5	32.4	0.0	0.0	8.8	23.6	9.1	24.5	0.0	0.0
LS-100 / LS-360	A1	PL [23]	L0	10.5	25.2	10.7	25.8	45.3	54.5	7.3	18.7	7.6	19.4	39.2	52.4
	A2	PL [23]	L0 + LM	9.1	23.4	9.4	23.9	66.5	70.0	7.0	18.2	7.1	18.6	51.3	61.4
	A3	MPL	L0	9.4	22.4	9.6	22.6	63.6	81.2	7.0	17.2	7.2	17.5	48.7	72.3
	A4	MPL	A2†	**8.7**	**22.0**	**9.0**	**22.4**	**72.2**	**83.0**	**6.5**	**16.9**	**6.8**	**17.1**	**58.5**	**76.4**
	A5	topline	L0	6.7	20.2	7.3	20.3	100.0	100.0	4.8	15.0	5.1	14.9	100.0	100.0
LS-100 / LS-860	B1	PL [23]	L0	10.4	24.3	10.7	25.0	37.0	41.4	7.3	18.0	7.4	18.4	34.4	43.5
	B2	PL [23]	L0 + LM	8.7	21.4	8.9	21.9	59.4	58.5	6.6	16.6	6.9	17.0	45.6	52.8
	B3	MPL	L0	9.0	18.0	9.2	18.1	56.0	80.0	6.9	13.7	7.0	14.1	44.2	73.7
	B4	MPL	B2†	**8.2**	**17.5**	**8.4**	**17.6**	**66.2**	**83.0**	**6.3**	**13.5**	**6.4**	**13.7**	**55.0**	**76.8**
	B5	topline	L0	5.7	14.4	5.8	14.6	100.0	100.0	4.1	10.5	4.3	10.4	100.0	100.0

†We used the model from 100 epochs and applied MPL for 100 epochs so that the total number of training epochs matches that of the other methods.

Table 2: *WER [%] and WRR [%↑] on out-domain TEDLIUM3 (TED3) setting.*

				Decoding without LM			Decoding with LM		
Setting	**Method**		**Init.**	**Dev WER** [%]	**Test WER** [%]	**Test WRR** [%↑]	**Dev WER** [%]	**Test WER** [%]	**Test WRR** [%↑]
LS-100	L0	baseline	–	32.5	33.2	0.0	26.8	26.8	0.0
LS-100 / TED3	C1	PL [23]	L0	26.8	26.0	35.6	22.1	20.9	34.6
	C2	PL [23]	L0 + LM	24.4	23.2	49.1	21.0	20.1	39.2
	C3	MPL	L0	18.8	17.6	76.3	16.8	15.5	65.9
	C4	MPL	C2	**18.2**	**16.7**	**80.8**	**16.2**	**14.9**	**69.6**
	C5	topline	L0	12.7	12.8	100.0	10.0	9.6	100.0

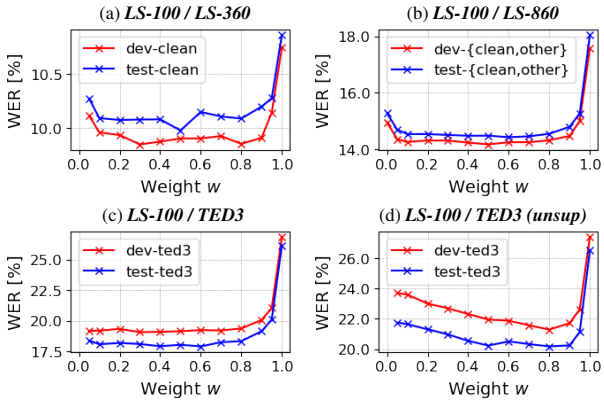

Figure 1: *Influence of momentum update weight w on WER.*

to the results with SpecAugment (A3). Note that, for the experiment without SpecAugment, the WRRs were calculated based on a topline model trained without the augmentation. Without SpecAugment, we observed that the MPL training converged earlier and MPL was less effective.

3.5. Effectiveness of w for tuning the momentum update

Figure 1 shows the model performance depending on the weight w used to calculate α in the momentum update (5). Here, we observed a similar trend among the curves in different semi-supervised settings (Figs. 1(a), 1(b), and 1(c)). The performance degraded as w was set closer to 0.0. When $w = 0.0$, which is a similar approach to [26], the training was likely to be unstable and failed under the conditions in Figs. 1(a) and 1(c), indicating the importance of retaining parameters from the base model. However, depending too much on the base model (i.e., setting w closer to 1.0) also worsened performance, since larger w slows down the offline model's progress, causing MPL to become more like standard PL [23]. Fig. 1(d) shows results under an extreme condition, where not only a domain mismatch exists between the base model and unlabeled data but labeled data is not used during the semi-supervised process (i.e., training with (4) only). The performance was more sensitive to the change in w than in the other settings, but the overall trend was similar.

Overall, the proposed tuning method effectively controlled the momentum update in all settings. It provides a more intuitive guide for tuning α, taking the amount of data into account. Based on the validation results mainly on the LS-860 and TED3 settings, we set $w = 0.5$ for all the semi-supervised conditions.

Table 3: *Comparison of test WER [%] between iterative PL and MPL in each setting. PL# denotes PL at the #-th iteration.*

Setting	**Test data**	**PL1**	**PL2**	**PL3**	**PL4**	**MPL**
LS-100 / LS-360	test-clean	11.3	10.5	10.2	9.9	**9.6**
	test-other	27.3	25.5	24.6	24.2	**22.6**
LS-100 / LS-860	test-clean	11.1	10.3	9.7	9.4	**9.2**
	test-other	25.6	23.2	21.9	20.7	**18.1**
LS-100 / TED3	test-ted3	26.3	23.5	22.2	21.3	**17.6**

3.6. Comparison with iterative pseudo-labeling

In Table 3, we list results comparing MPL with a simple iterative PL (IPL) method similar to [25, 30]. In IPL, the base model was continuously trained on pseudo-labels which were updated at intervals of 50 epochs. For a fair comparison, the update was performed four times to match the total number of epochs to that of MPL (i.e., 200 epochs). The results show the effectiveness of IPL, as performance gradually improved with the iterations. However, MPL still outperformed the final round of PL, indicating MPL is more effective at training with unlabeled data.

4. Conclusions

This paper proposed MPL, momentum pseudo-labeling for semi-supervised ASR. Experimental results on various semi-supervised settings demonstrated its effectiveness, achieving clear improvements over standard pseudo-labeling and iterative pseudo-labeling. Moreover, MPL was shown to be effective independently of the amount of unlabeled data or domain mismatch. Future work includes applying filtering techniques [31] and introducing multiple hypotheses [32] in the MPL process.

5. References

[1] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Process. Mag.*, vol. 29, no. 6, 2012.

[2] A. Graves, A. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. ICASSP*, 2013.

[3] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *Proc. ICML*, 2014.

[4] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *Proc. NeurIPS*, 2015.

[5] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *Proc. ICASSP*, 2016.

[6] C.-C. Chiu, T. N. Sainath, Y. Wu, R. Prabhavalkar, P. Nguyen, Z. Chen, A. Kannan *et al.*, "State-of-the-art speech recognition with sequence-to-sequence models," in *Proc. ICASSP*, 2018.

[7] C. Lüscher, E. Beck, K. Irie, M. Kitza, W. Michel, A. Zeyer, R. Schlüter *et al.*, "RWTH ASR systems for LibriSpeech: Hybrid vs attention," in *Proc. Interspeech*, 2019.

[8] S. Karita, N. Chen, T. Hayashi, T. Hori, H. Inaguma, Z. Jiang, M. Someki *et al.*, "A comparative study on Transformer vs RNN in speech applications," in *Proc. ASRU*, 2019.

[9] A. Tjandra, S. Sakti, and S. Nakamura, "Listening while speaking: Speech chain by deep learning," in *Proc. ASRU*, 2017.

[10] T. Hori, R. Astudillo, T. Hayashi, Y. Zhang, S. Watanabe, and J. Le Roux, "Cycle-consistency training for end-to-end speech recognition," in *Proc. ICASSP*, 2019.

[11] M. K. Baskar, S. Watanabe, R. Astudillo, T. Hori, L. Burget, and J. Černocký, "Semi-supervised sequence-to-sequence ASR using unpaired speech and text," in *Proc. Interspeech*, 2019.

[12] S. Karita, S. Watanabe, T. Iwata, A. Ogawa, and M. Delcroix, "Semi-supervised end-to-end speech recognition," in *Proc. Interspeech*, 2018.

[13] J. Drexler and J. Glass, "Combining end-to-end and adversarial training for low-resource speech recognition," in *Proc. SLT*, 2018.

[14] Y. Ren, X. Tan, T. Qin, S. Zhao, Z. Zhao, and T.-Y. Liu, "Almost unsupervised text to speech and automatic speech recognition," in *Proc. ICML*, 2019.

[15] A. Baevski, M. Auli, and A. Mohamed, "Effectiveness of self-supervised pre-training for speech recognition," *arXiv preprint arXiv:1911.03912*, 2019.

[16] S. Ling, Y. Liu, J. Salazar, and K. Kirchhoff, "Deep contextualized acoustic representations for semi-supervised speech recognition," in *Proc. ICASSP*, 2020.

[17] S. Schneider, A. Baevski, R. Collobert, and M. Auli, "wav2vec: Unsupervised pre-training for speech recognition," in *Proc. Interspeech*, 2019.

[18] Y.-A. Chung and J. Glass, "Generative pre-training for speech with autoregressive predictive coding," in *Proc. ICASSP*, 2020.

[19] A. Baevski, S. Schneider, and M. Auli, "vq-wav2vec: Self-supervised learning of discrete speech representations," in *Proc. ICLR*, 2019.

[20] W.-N. Hsu, Y.-H. H. Tsai, B. Bolte, R. Salakhutdinov, and A. Mohamed, "HUBERT: How much can a bad teacher benefit ASR pre-training," in *Proc. ICASSP*, 2021.

[21] H. Scudder, "Probability of error of some adaptive pattern-recognition machines," *IEEE Trans. Inf. Theory*, vol. 11, no. 3, 1965.

[22] D.-H. Lee, "Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks," in *Proc. ICML Workshop on Challenges in Representation Learning*, 2013.

[23] J. Kahn, A. Lee, and A. Hannun, "Self-training for end-to-end speech recognition," in *Proc. ICASSP*, 2020.

[24] R. Masumura, M. Ihori, A. Takashima, T. Moriya, A. Ando, and Y. Shinohara, "Sequence-level consistency training for semi-supervised end-to-end automatic speech recognition," in *Proc. ICASSP*, 2020.

[25] Q. Xu, T. Likhomanenko, J. Kahn, A. Hannun, G. Synnaeve, and R. Collobert, "Iterative pseudo-labeling for speech recognition," in *Proc. Interspeech*, 2020.

[26] Y. Chen, W. Wang, and C. Wang, "Semi-supervised ASR by end-to-end self-training," in *Proc. Interspeech*, 2020.

[27] D. S. Park, Y. Zhang, Y. Jia, W. Han, C.-C. Chiu, B. Li, Y. Wu *et al.*, "Improved noisy student training for automatic speech recognition," in *Proc. Interspeech*, 2020.

[28] F. Weninger, F. Mana, R. Gemello, J. Andrés-Ferrer, and P. Zhan, "Semi-supervised learning with data augmentation for end-to-end ASR," in *Proc. Interspeech*, 2020.

[29] W.-N. Hsu, A. Lee, G. Synnaeve, and A. Hannun, "Semi-supervised speech recognition via local prior matching," *arXiv preprint arXiv:2002.10336*, 2020.

[30] T. Likhomanenko, Q. Xu, J. Kahn, G. Synnaeve, and R. Collobert, "slimIPL: Language-model-free iterative pseudo-labeling," *arXiv preprint arXiv:2010.11524*, 2020.

[31] S. Khurana, N. Moritz, T. Hori, and J. Le Roux, "Unsupervised domain adaptation for speech recognition via uncertainty driven self-training," in *Proc. ICASSP*, 2021.

[32] N. Moritz, T. Hori, and J. Le Roux, "Semi-supervised speech recognition via graph-based temporal classification," in *Proc. ICASSP*, 2021.

[33] S. Laine and T. Aila, "Temporal ensembling for semi-supervised learning," in *Proc. ICLR*, 2017.

[34] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," in *Proc. NeurIPS*, 2017.

[35] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *Proc. ICML*, 2006.

[36] S. Kriman, S. Beliaev, B. Ginsburg, J. Huang, O. Kuchaiev, V. Lavrukhin, R. Leary *et al.*, "QuartzNet: Deep automatic speech recognition with 1D time-channel separable convolutions," in *Proc. ICASSP*, 2020.

[37] Y. Higuchi, S. Watanabe, N. Chen, T. Ogawa, and T. Kobayashi, "Mask CTC: Non-autoregressive end-to-end ASR with CTC and mask predict," *Proc. Interspeech*, 2020.

[38] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A simple data augmentation method for automatic speech recognition," in *Proc. Interspeech*, 2019.

[39] R. Takashima, S. Li, and H. Kawai, "An investigation of a knowledge distillation method for CTC acoustic models," in *Proc. ICASSP*, 2018.

[40] Q. Xie, Z. Dai, E. Hovy, M.-T. Luong, and Q. V. Le, "Unsupervised data augmentation for consistency training," in *Proc. NeurIPS*, 2020.

[41] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in *Proc. ICASSP*, 2015.

[42] F. Hernandez, V. Nguyen, S. Ghannay, N. Tomashenko, and Y. Estève, "TED-LIUM 3: Twice as much data and corpus repartition for experiments on speaker adaptation," in *Proc. SPECOM*, 2018.

[43] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann *et al.*, "The Kaldi speech recognition toolkit," in *Proc. ASRU*, 2011.

[44] T. Kudo, "Subword regularization: Improving neural network translation models with multiple subword candidates," in *Proc. ACL*, 2018.

[45] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N. Enrique Yalta Soplin *et al.*, "ESPnet: End-to-end speech processing toolkit," in *Proc. Interspeech*, 2018.

[46] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser *et al.*, "Attention is all you need," in *Proc. NeurIPS*, 2017.

[47] 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," in *Proc. Interspeech*, 2019.

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

[49] J. Ma and R. Schwartz, "Unsupervised versus supervised training of acoustic models," in *Proc. Interspeech*, 2008.