Momentum Pseudo-Labeling for Semi-Supervised Speech Recognition

Yosuke Higuchi\textsuperscript{1,2*}, Niko Moritz\textsuperscript{1}, Jonathan Le Roux\textsuperscript{1}, Takaaki Hori\textsuperscript{1}

\textsuperscript{1}Mitsubishi Electric Research Laboratories (MERL), USA
\textsuperscript{2}Waseda University, Japan

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) \cite{1,2}. Much of the recent progress lies in the end-to-end (E2E) framework \cite{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 \cite{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 \cite{9–11} or a sequential auto-encoder \cite{12–14}. Others adopted self-supervised pre-training techniques, such as BERT-like mask prediction \cite{15,16}, contrastive loss \cite{17,18}, and feature clustering \cite{19,20}, to promote downstream E2E ASR tasks.

We focus on self-training \cite{21} or pseudo-labeling (PL) \cite{22}, which has recently been adopted for semi-supervised E2E ASR and shown to be effective \cite{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 \cite{23,29}. Data augmentation is also useful for helping the student model with training on the pseudo-labeled data \cite{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 \cite{25,26,30,31}. In \cite{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 \cite{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 \cite{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 \cite{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:

\textbf{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.

\textbf{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 \cite{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.
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 \( X = (x_t) \in \mathbb{R}^T \) and an L-length output sequence \( Y = (y_t) \in \mathbb{V}^L \). Here, \( x_t \) is a D-dimensional acoustic feature at frame \( t \), \( y_t \) an output token at position \( l \), and \( \mathbb{V} \) 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 an output token at position \( t \) given an 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 \( D_{sup} \) is available during the semi-supervised process, the supervised loss \( \mathcal{L}_{sup}(\theta) \) can be incorporated into the training, helping stabilize the online model as it learns from unlabeled data with \( \mathcal{L}_{unsup}(\xi) \). 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

\[
\phi \leftarrow \alpha \phi + (1 - \alpha) \xi,
\]

(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 \( \alpha \) in (5), we design a novel, more intuitive method for deriving an appropriate value of \( \alpha \). Based on (5), the parameters of the offline model after \( K \) iterations can be computed as follows:

\[
\phi^{(K)} = \alpha^K \phi^{(0)} + (1 - \alpha) \sum_{k=0}^{K-1} \alpha^{K-k} \xi^{(k)},
\]

(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 hence 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 \( \alpha \) lead to huge differences in \( w \) (e.g., \( 0.999^{3000} \approx 0.9997^{3000} \)), requiring small adjustments on \( \alpha \) for different amounts of training data. Instead of directly tuning \( \alpha \) 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 \( K \), \( \alpha \) is calculated as \( \alpha = \phi^{(K)} / \log w \). 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]. The base model was trained for 150 epochs using the Adam optimizer [48] with $\beta_1 = 0.9, \beta_2 = 0.98, \epsilon = 10^{-8}$, 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/LS-360, $\alpha = 0.99989$ for LS-100/LS-860, and $\alpha = 0.99983$ for LS-100/TED3.

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 mismatch, 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...
Figure 1 shows the model performance depending on the weight $w$ earlier and MPL was less effective. Without SpecAugment, the WRRs were calculated based to the results with SpecAugment ($\lambda_3$). 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 $\alpha$ 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 $\alpha$, 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.

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 training epochs that match the other methods. 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 hypothesizes [32] in the MPL process.
S. References


