A PRELIMINARY STUDY ON ASSOCIATED LEARNING FOR ASR

Pin-Jui Ku[*1], *Phani Sankar Nidadavolu*[2], *Brian King*[2], *Pegah Ghahremani*[2], *I-Fan Chen*[2]

[1]School of ECE, Georgia Institute of Technology
[2]Amazon Inc.

ABSTRACT

In this paper, we propose the first successful implementation of associated learning (AL) to automatic speech recognition (ASR). AL has been shown to provide better label noise robustness, faster training convergence, and flexibility on model complexity than back-propagation (BP) in classification tasks. However, extending the learning approach to autoregressive models such as ASR, where model outputs are part of inputs for future prediction, is challenging. We solve the problem by applying AL at the token level and propose an updated AL loss to ensure the correct learning behavior. Experimental results on the TIMIT task show that the proposed AL models achieve the same phoneme accuracy as the baseline BP models with 15% fewer model parameters.

Index Terms— associated learning, back-propagation, machine learning, representation learning

1. INTRODUCTION

The broader usage of Automatic speech recognition (ASR) brings up several challenges to modern ASR systems. On the data side, modern ASR systems need to be able to handle different types of audio conditions [1, 2, 3] and be robust to training label errors [4]. On the maintainability and scalability side, it is preferable to have a flexible model architecture that allows easy adjustment on inference computation cost and memory footprint with marginal accuracy degradation, so that a single model may serve different use cases (e.g., in the cloud or on the device) with different computation constraints [5, 6]. The model should also be able to easily expand its capacity to cover new domain use cases without degrading performance on existing domains [7, 8]. Recently, the machine learning community has presented associated learning (AL) [9] as an alternative to conventional back-propagation (BP) [10] for model training, which has some properties that may serve as a unified framework to address all the challenges mentioned above.

In general neural network models, the models transform input features into output labels. The transformations are learned by back-propagating the gradients from loss functions between the predicted and target labels. Neural network models with this training approach are named BP form in [9]. AL views the relationship between features and labels differently. Rather than transformations, it learns the "association" between input features and output labels in representation spaces. To apply AL, the BP form network must be converted into AL form [9]. In [9], the authors proposed AL forms for several model architectures used in classification tasks. They also show that AL provides better robustness to training label errors, faster training-converge time, properties allowing more efficient training parallelization, and flexibility on model complexity. These properties can also help address the challenges modern ASR systems face and thus motivate us to explore applying AL to ASR.

The experiments in [9] are limited to classification tasks. It is more challenging to extend the learning approach to autoregressive models such as ASR, where model outputs are also inputs for future prediction. In this preliminary study, we focus on developing a functional AL form for a Listen, Attend, and Spell (LAS) model architecture, where the input-output alignments are introduced via cross-attention between encoder-decoder. To simplify the problem, we choose the TIMIT phoneme recognition task [11] as the task for our study. We believe the findings can be extended to general ASR tasks, and consider them as our future work. To the authors' knowledge, this paper is the first successful attempt to apply AL to ASR models. The contributions of the paper are as follows: (1) we propose the first functional AL forms for the LAS phoneme recognizer, (2) we present preliminary analyses on AL for the LAS phoneme recognizer, and (3) we identify the challenges of applying AL to autoregressive models and provide suggestions on how the challenges may be resolved.

2. RELATION TO PRIOR WORK

The concept of associated learning aligns with several techniques used in the ASR community. For example, in representation learning, contrastive predictive coding (CPC)-based approaches [12, 13] learn a representation space in an unsupervised manner that captures the information required for ASR tasks only using acoustic data. This is, however, different from AL, which learns the representation space from

*The first author performed the work during their internship at Amazon.

labeled data; AL also tends to find a representation space shared among model inputs and outputs rather than just for the inputs.

3. METHODS

3.1. Baseline BP form LAS Model

Figure 1 illustrates the baseline BP form model architecture. The baseline model is an LAS model [14] based on the implementation in [15]. The encoder consists of N_E bidirectional gated recurrent unit (GRU) layers which transform the sequence of input acoustic features $\{\overrightarrow{x}_i\}_{i=1}^{T}$ into hidden state vectors $\mathbf{h} = \{\overrightarrow{h}_i\}_{i=1}^{T}$, where T is the number of feature frames in the sequence. The autoregressive decoder consists of N_D layers of GRU, whose inputs are previous labels $\{y_k\}_{k=1}^{j-1}$ when predicting the jth output token. On top of the GRU layers, a cross-attention module over the encoder hidden state vectors $\mathbf{h}$ followed by a feed-forward network are added for generating the output tokens. The first output token y_1 is always $\langle sos \rangle$ and the last output token y_U is always $\langle eos \rangle$ representing the start and end of sentence tokens respectively, where U is the length of the output sequence.

The feed-forward path in the decoder consists of a sequence of transformation $\boldsymbol{f}^1$, $\boldsymbol{f}^2$, ..., $\boldsymbol{f}^{N_D}$, cross-attention module, and the final feed-forward (FF) network. The model is trained by back-propagating the loss, e.g., cross-entropy (CE) loss, from the output back to each transformation component using the chain rule.

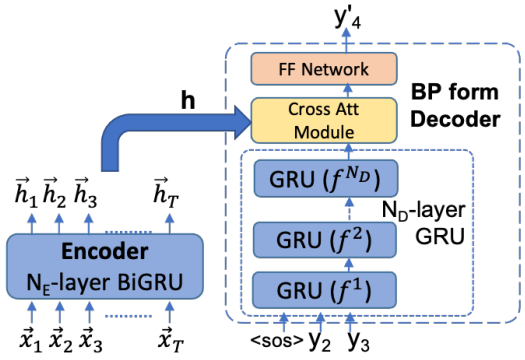

Fig. 1. The baseline LAS model architecture, where the decoder is in BP form. Here the decoder is generating the 4th output token y'_4 based on the history.

3.2. Associated Learning for LAS models

To apply AL, we must convert the BP form network into AL form [9]. The key is to introduce three projections to the model: $\boldsymbol{b}$, a bridge function that projects hidden vectors to the representation space; $\boldsymbol{g}$, a label encoder function that projects the label to the representation space; and $\boldsymbol{h}$, a decoder function that reconstructs the label from a representation vector.

In this work, we explore applying AL to the LAS-based phoneme recognition models. We focus on deriving the AL form for the LAS decoder. Figure 2 shows the proposed AL form for the LAS decoder. When comparing Figure 2 with Figure 1, we can see both decoder structures have the same GRU transformations ($\boldsymbol{f}^1$, $\boldsymbol{f}^2$, ..., $\boldsymbol{f}^{N_D}$). However, the rest of the model architecture is quite different with the additional $\boldsymbol{b}^i$, $\boldsymbol{g}^i$, and $\boldsymbol{h}^i$ components as explained below.

On the input side, for each GRU layer $\boldsymbol{f}^i$, a bridge function component $\boldsymbol{b}^i$ is introduced to transform the hidden vector output from $\boldsymbol{f}^i$ into a vector $\overrightarrow{s}_j^i$ in the ith representation space for the jth output token. We implemented the bridge function $\boldsymbol{b}^i$ using a cross-attention module which takes the hidden vector output from $\boldsymbol{f}^i$ as a query and attends to the encoder hidden state vectors $\mathbf{h}$.

On the output token side, we introduce label encoder functions $\boldsymbol{g}^i$, which are simple feed-forward networks, stacked together to project the target next token, y'_j, to the ith representation space as representation vectors $\overrightarrow{t}_j^i$. The label decoder functions $\boldsymbol{h}^i$, which is another set of feed-forward networks are introduced to reconstruct the label representation vector at ith layer, $\overrightarrow{t}_j^i$, back to $(i-1)$th layer, $\overrightarrow{t}_j^{i-1}$, and eventually y'_j.

3.2.1. *Training the AL form LAS models*

We train the proposed AL form LAS models using teacher forcing. For the jth target token, we compute its corresponding ith layer representation vector, $\overrightarrow{s}_j^i$, based on acoustic encoder hidden states ($\mathbf{h}$) and token history $\{y_k\}_1^{j-1}$ via $\boldsymbol{f}^i$ and $\boldsymbol{b}^i$ functions. We also compute the ith layer token representation vector, t_j^i, using $\boldsymbol{g}^i$ functions. In addition, we would like to learn the $\boldsymbol{h}^i$ function reconstructing $\overrightarrow{t}_j^{i-1}$ from $\overrightarrow{t}_j^i$. The loss function at the ith layer ($i \geq 2$) is, therefore:

$$Loss^i = MSE(\overrightarrow{s}_j^i, \overrightarrow{t}_j^i) + MSE(\overrightarrow{t}_j^{i-1}, \boldsymbol{h}^i(\overrightarrow{t}_j^i)), \quad (1)$$

where the first mean squared error (MSE) loss is the associated loss encouraging the representation vectors $\overrightarrow{s}_j^i$ and $\overrightarrow{t}_j^i$ to be as close as possible; while the second MSE loss is the reconstruction loss to ensure $\overrightarrow{t}_j^{i-1} = \boldsymbol{h}^i(\boldsymbol{g}^i(\overrightarrow{t}_j^{i-1})) \approx \overrightarrow{t}_j^{i-1}$. When $i = 1$, $y'_j = \boldsymbol{h}^1(\overrightarrow{t}_j^1)$, we use CE as the reconstruction loss, i.e.,

$$Loss^1 = MSE(\overrightarrow{s}_j^1, \overrightarrow{t}_j^1) + CE(y_j, \boldsymbol{h}^1(\overrightarrow{t}_j^1)). \quad (2)$$

Combining Eq. 1 and 2, the overall loss in training is then

$$Overall\ Loss = \sum_{i=1}^{N_D} Loss^i. \quad (3)$$

From Eq. 1 and 2, we can see the loss for layer i can be used as the local loss for $\boldsymbol{f}^i$, $\boldsymbol{b}^i$, $\boldsymbol{g}^i$, and $\boldsymbol{h}^i$ training, i.e., there

is no dependency from the other layers. This property allows better training parallelization for AL form models [9]. With this, we follow the implementation in [9] and apply a gradient stop between each layer in our AL form LAS decoding during training. We also empirically confirmed the findings in [9] that no significant gain was observed when enabling gradient flow between layers.

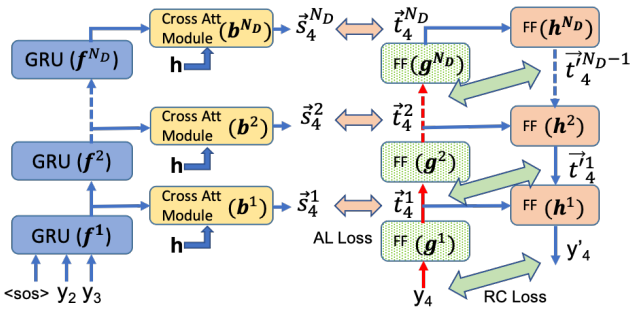

Fig. 2. The proposed AL form LAS decoder is used to replace the BP form decoder shown in Figure 1. The bridge function $\boldsymbol{b}^i$ at the ith layer is implemented by a cross-attention module; the label encoder and decoder functions $\boldsymbol{g}^i$ and $\boldsymbol{h}^i$ are feed-forward (FF) networks. The model is trained with the MSE loss between $\overrightarrow{s}_j^i$ and $\overrightarrow{t}_j^i$ as the AL loss and the reconstruction (RC) loss between $\overrightarrow{t}_j^i$ and $\overrightarrow{t'}_j^i$.

3.2.2. Decoding with AL form LAS models

The label encoder components $\boldsymbol{g}^i$ shown in Figure 2 are only used at training time. At inference time, we drop all of them from the model. The full inference path in the AL form LAS decoder for the next output token y'_j with the decoding history $\{y'_k\}_{k=1}^{j-1}$ and the acoustic encoder hidden states vectors $\mathbf{h}$ are described below. Similar to the BP form, the decoding history $\{y'_k\}_{k=1}^{j-1}$ is fed into the GRU layers $\boldsymbol{f}^1$ to $\boldsymbol{f}^{N_D}$. The output of the last GRU layer $\boldsymbol{f}^{N_D}$ is then transformed to $\overrightarrow{s}_j^{N_D}$ using the cross-attention module based bridge function $\boldsymbol{b}^{N_D}$. Since during the AL training, the model learns to bring the projection of $\overrightarrow{s}_j^i$ and $\overrightarrow{t}_j^i$ close in the ith layer representation space, we can replace $\overrightarrow{t}_j^{N_D}$ with $\overrightarrow{s}_j^{N_D}$ and feed the vector into the label decoder component $\boldsymbol{h}^{N_D}$ to reconstruct $\overrightarrow{t}_j^{N_D-1}$ as $\overrightarrow{t'}_j^{N_D-1}$. We then continue to feed the reconstructed $\overrightarrow{t'}_j^{N_D-1}$ into $\boldsymbol{h}^{N_D-1}$ to reconstruct the label representation vector at the lower layer until y'_j is reconstructed. The rest of the decoding process is the same as the BP form LAS model.

The inference path referenced above is the full inference path, meaning that we use all of the $\boldsymbol{f}^i$ and $\boldsymbol{h}^i$ components. However, there are also shortcut paths available for inference [9]. For example, we may just use the first GRU layer with the bridge function $\boldsymbol{b}^1$ to create $\overrightarrow{s}_j^1$ and reconstruct y'_j accordingly using $\boldsymbol{h}^1$. These shortcut paths provide the flexibility to control the required computation at inference time without significantly impacting model accuracy.

3.3. Associated Learning With Variational Autoencoder

Section 3.2 describes the AL form decoder designed following [9]. However, we found the proposed AL form LAS model has a very high phoneme error rate (PER) (see Table 1). Our investigation shows the AL models tend to shrink the norm of the hidden representations ($\overrightarrow{s}_j^i$, $\overrightarrow{t}_j^i$, and $\overrightarrow{t'}_j^i$) down near zero to reduce the MSE losses in Eq. 1 without learning meaningful transformation. We also find the trained models only rely on the previous phonemes, $\{y_k\}_{k=1}^{j-1}$, for the next token prediction without paying attention to $\mathbf{h}$. The observations show the AL loss is insufficient for the models to learn a proper association between input and outputs. Applying batch norm or layer norm or in training does not help.

Table 1. The PER (%) on TIMIT Dev and Test sets for the baseline BP form and AL form LAS models when $N_D = 1$.

Model	Learning Criterion	Dev	Test
BP	Back propagation	24.5	25.7
AL	AL	>100	>100
AL	AL with VAE loss	20.5	21.8

To address the issue, we replace the feed-forward networks in the label encoder function $\boldsymbol{g}^i$ with variational autoencoder (VAE) components so that the t_j^i becomes

$$\overrightarrow{t}_j^i = \overrightarrow{\mu}^i + \overrightarrow{\epsilon} \cdot e^{\overrightarrow{v}^i/2}, \tag{4}$$

where $\overrightarrow{\mu}^i$ and $\overrightarrow{v}^i$ are the estimated mean and log variance vectors, and $\overrightarrow{\epsilon}$ is a noise vector sampled from standard distribution. To train the component, we add a modified VAE loss term $e^{\overrightarrow{v}^i} - (1 + \overrightarrow{v}^i)$, which is the Kullback–Leibler divergence (KLD) loss term without the mean part, to Eq. 1 and 2. We remove the mean part on purpose to avoid the shrinking issue in the AL setup. The noise term $\overrightarrow{\epsilon}^i \cdot e^{\overrightarrow{v}^i}$ in the updated $\overrightarrow{t}_j^i$ (Eq. 4) forces the vector to stay away from zero so the models learn the proper association between $\overrightarrow{s}_j^i$ and $\overrightarrow{t}_j^i$ better with MSE losses.

4. EXPERIMENTS

4.1. Experimental Setup

We verify the proposed AL form LAS implementation with the TIMIT [11] phoneme recognition task. The baseline system is based on the implementation in [15] except that, for simplicity, we use the single head attention in the cross-attention module and remove the context feedback to decoder

Table 2. PER(%) comparison between BP and AL forms when $N_D \geq 1$. BP (N_D) means the BP form baseline model with N_D decoder layers. AL (N_D, s) denotes the decoder layer depth at training time (N_D) and inference time (s) for the AL models.

Model	# of layers in decoder	# parameters (M) Train	# parameters (M) Inference	Inference layer	PER - Train Mean ± STD	PER - Dev Mean ± STD	PER - Test Mean ± STD
BP (1)	1	4.0	4.0	-	4.3 ± 1.9	24.5 ± 1.4	25.7 ± 1.6
BP (2)	2	4.4	4.4	-	2.5 ± 0.8	20.5 ± 0.1	22.1 ± 0.5
BP (3)	3	4.8	4.8	-	3.3 ± 0.8	20.4 ± 0.4	**21.8** ± 0.4
AL (1, 1)	1	4.4	4.1	1	7.7 ± 0.6	20.5 ± 0.2	**21.8** ± 0.2
AL (2, 1)	2	5.7	4.1	1	5.3 ± 0.6	**20.2** ± 0.4	**21.8** ± 0.5
AL (2, 2)	2	5.7	4.6	2	5.8 ± 0.6	20.5 ± 0.2	22.1 ± 0.4
AL (3, 1)	3	7.0	4.1	1	4.1 ± 0.5	20.6 ± 0.2	22.2 ± 0.3
AL (3, 2)	3	7.0	4.6	2	4.4 ± 0.6	20.9 ± 0.3	22.8 ± 0.4
AL (3, 3)	3	7.0	5.2	3	5.3 ± 0.7	22.2 ± 0.3	24.0 ± 0.4

input. All LAS models use the same encoder structure with three bidirectional GRU layers ($N_E = 3$) and 256 for hidden size. The encoder input is the 80-dimensional Mel filter bank features stacked by three, i.e. a total feature size of 240. Both BP and AL form models are trained with 0.5 dropout rate. The decoder input and output are the 61 TIMIT phonemes, plus $\langle sos \rangle$, $\langle eos \rangle$, and the padding token. We map the 61 TIMIT phonemes to 39 phonemes before the phoneme error rate (PER) computation [15]. All the results are reported in the PER metric. We run each experiment five times and report the PER mean and standard deviation.

4.2. Single Decoder Layer ($N_D = 1$) Results

We start by testing the proposed AL form LAS model with a single decoder layer. Table 1 shows the PERs of the baseline BP form and the proposed AL form LAS models. The AL model has very high PERs (>100%) on both Dev and Test sets as the model just repeats the same phoneme for output as discussed in section 3.3. The VAE loss proposed in section 3.3 resolves the problem. The AL with VAE loss system achieves PER at 20.5% and 21.8% on Dev and Test. The PERs are 15% relative lower than the baseline BP form at 24.5% and 25.7% for Dev and Test sets, respectively. For the rest of the experiments, the "AL model" refers to the "AL with VAE model".

4.3. Multiple Decoder Layers ($N_D \geq 1$) Results

Table 2 shows the full results with $N_D \geq 1$ for both BP and AL form models. The table also shows the number of parameters at training and inference time. The BP form models have the same parameter number at both training and inference time. However, for AL form models, since we do not need label encoders, $\boldsymbol{g}^i$, for inference, as described in section 3.2.2, the number of model parameters at inference time are smaller than the training time. With the multiple inference path options for AL, we can further reduce the inference time parameter for AL form models.

In Table 2, the baseline BP models have lower PER as the number of decoder layers, N_D, increases. When $N_D = 3$, the BP model achieves the best average PER at 21.8% with 4.8M inference-time parameters. On the other hand, the AL form models, AL(1,1) and AL(2,1), achieves the same PER at 21.8% with 15% fewer interference-time parameters (i.e., 4.8M to 4.1M). We find that increasing N_D for AL form models, though reducing the training PER, does not help further reduce PER for AL form models. This is probably due to the small training data size of the TIMIT task. We also observe that, though the proposed VAE variation mitigates the problem that AL form models rely on the previous token for prediction, the training label sequence over-fitting issue seems to remain to some extent. When examining the results in Table 2 for AL form with $N_D = 3$, we observed the AL models tend to remember the training label sequence more when N_D increases. This degrades PER on all data sets including training data. Reducing such training label sequence overfitting issues in AL form will be one of our future research directions.

5. CONCLUSION

In this paper, we propose the first successful implementation of the AL form LAS model to the TIMIT phoneme recognition task. We observe a VAE variation of the AL loss is required to ensure correct learning for autoregressive models. Experimental results on the TIMIT task show AL models have flexible inference paths and can achieve the same phoneme accuracy as the baseline BP models with 15% fewer model parameters. For future work, we will extend the approach to end-to-end ASR tasks such as LibriSpeech and explore more complicated model architectures, e.g., LAS with context feedback and RNN-T.

6. REFERENCES

[1] Jinyu Li, Li Deng, Yifan Gong, and Reinhold Haeb-Umbach, "An Overview of Noise-Robust Automatic Speech Recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 4, pp. 745–777, Apr. 2014.

[2] Shane Settle, Jonathan Le Roux, Takaaki Hori, Shinji Watanabe, and John R. Hershey, "End-to-End Multi-Speaker Speech Recognition," in *Proc. ICASSP*, Apr. 2018, pp. 4819–4823.

[3] Tobias Menne, Ilya Sklyar, Ralf Schlüter, and Hermann Ney, "Analysis of Deep Clustering as Preprocessing for Automatic Speech Recognition of Sparsely Overlapping Speech," in *Proc. Interspeech*, Sept. 2019, pp. 2638–2642.

[4] I-Fan Chen, Brian King, and Jasha Droppo, "Investigation of Training Label Error Impact on RNN-T," `http://arxiv.org/abs/2112.00350`, Dec. 2021.

[5] Bhuvana Ramabhadran, Kartik Audhkhasi, Pedro Jose Moreno Mengibar, and Tongzhou Chen, "Mixture model attention: Flexible streaming and non-streaming automatic speech recognition," in *Proc. of Interspeech*, 2021.

[6] Aleksei Kalinov, Somshubra Majumdar, Jagadeesh Balam, and Boris Ginsburg, "Carnelinet: Neural mixture model for automatic speech recognition," `https://arxiv.org/abs/2107.10708`, 2021.

[7] Katrin Tomanek, Vicky Zayats, Dirk Padfield, Kara Vaillancourt, and Fadi Biadsy, "Residual Adapters for Parameter-Efficient ASR Adaptation to Atypical and Accented Speech," `https://arxiv.org/abs/2109.06952`, Sept. 2021.

[8] Heng-Jui Chang, Hung-yi Lee, and Lin-shan Lee, "Towards Lifelong Learning of End-to-end ASR," `https://arxiv.org/abs/2104.01616`, July 2021.

[9] Dennis YH Wu, Dinan Lin, Vincent Chen, and Hung-Hsuan Chen, "Associated learning: an alternative to end-to-end backpropagation that works on cnn, rnn, and transformer," in *Proc. International Conference on Learning Representations*, 2021.

[10] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, Oct. 1986.

[11] John S. Garofolo, Lori F. Lamel, William M. Fisher, David S. Pallett, Nancy L. Dahlgren, Victor Zue, and Jonathan G. Fiscus, "TIMIT Acoustic-Phonetic Continuous Speech Corpus," `https://catalog.ldc.upenn.edu/LDC93S1`, 1993.

[12] Yu-An Chung, Hao Tang, and James Glass, "Vector-quantized autoregressive predictive coding," `https://arxiv.org/abs/2005.08392`, 2020.

[13] Wei-Ning Hsu, Yao-Hung Hubert Tsai, Benjamin Bolte, Ruslan Salakhutdinov, and Abdelrahman Mohamed, "Hubert: How much can a bad teacher benefit asr pre-training?," in *Proc. ICASSP*, 2021, pp. 6533–6537.

[14] William Chan, Navdeep Jaitly, Quoc Le, and Oriol Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *Proc. ICASSP*, 2016, pp. 4960–4964.

[15] biyoml, "Pytorch-End-to-End-ASR-on-TIMIT," `https://github.com/biyoml/PyTorch-End-to-End-ASR-on-TIMIT`, Nov. 2021.