Hou *et al. EURASIP Journal on Audio, Speech, and Music Processing* (2020) 2020:3
https://doi.org/10.1186/s13636-020-0170-z

EURASIP Journal on Audio, Speech, and Music Processing

RESEARCH **Open Access**

Segment boundary detection directed attention for online end-to-end speech recognition

Junfeng Hou*, Wu Guo, Yan Song and Li-Rong Dai

Abstract

Attention-based encoder-decoder models have recently shown competitive performance for automatic speech recognition (ASR) compared to conventional ASR systems. However, how to employ attention models for online speech recognition still needs to be explored. Different from conventional attention models wherein the soft alignment is obtained by a pass over the entire input sequence, attention models for online recognition must learn online alignment to attend part of input sequence monotonically when generating output symbols. Based on the fact that every output symbol is corresponding to a segment of input sequence, we propose a new attention mechanism for learning online alignment by decomposing the conventional alignment into two parts: *segmentation*—segment boundary detection with hard decision—and *segment-directed attention*—information aggregation within the segment with soft attention. The boundary detection is conducted along the time axis from left to right, and a decision is made for each input frame about whether it is a segment boundary or not. When a boundary is detected, the decoder generates an output symbol by attending the inputs within the corresponding segment. With the proposed attention mechanism, online speech recognition can be realized. The experimental results on TIMIT and WSJ dataset show that our proposed attention mechanism achieves comparable online performance with state-of-the-art models.

Keywords: Encoder-decoder, Online recognition, Boundary detection, Attention mechanism, Reinforcement learning, Policy gradient

1 Introduction

With the booming of deep learning, attention-based encoder-decoder models have rapidly advanced the research of sequence transduction problems like machine translation, image caption, and automatic speech recognition (ASR) [1–6]. For a sequence transduction problem, the alignment between input and output sequence is essential. An effective solution for modeling the alignment is attention mechanism which can be learned jointly with the encoder-decoder model [1]. Generally, the attention-based encoder-decoder model consists of encoder, decoder, and attention mechanism. The encoder is a neural network such as recurrent neural network (RNN) [7–9] or transformer [10] which encodes each input item together with its context information into a hidden representation, and all the hidden representations of the input sequence constitute a memory bank. The decoder is a similar neural network which learns to generate output sequence given the encoder memory bank. And the attention mechanism iteratively performs a pass over the entire input memory to get attention weight vector which represents the soft alignment and can be used to aggregate information from the memory bank for decoder.

Although the conventional attention-based encoder-decoder models have proven to be effective on a wide variety of problems, they fail in streaming processing tasks like online speech recognition where output symbols are produced when the input sequence has only been partially observed. Therefore, the key for online recognition is to learn online alignment. Windowing approach was proposed in [4] where the range of the attention at current output step was restricted to a fixed-sized window with position decided by attention at previous output step. And it was adopted to online recognition in [11]. An

*Correspondence: hjf176@mail.ustc.edu.cn
National Engineering Laboratory for Speech and Language Information Processing, University of Science and Technology of China, Hefei 230022, China

extra inferring algorithm was introduced in [12] to split the input and output sequence into equally sized non-overlapping blocks over which the sequence-to-sequence transduction was performed. In [13], a jointly trained CTC-based network was introduced to the attention model to provide alignment information for triggering the online attention module. In [14, 15], the input speech was processed frame-by-frame to decide whether or not to emit an output symbol for each input. These decisions were optimized with reinforcement learning (RL), and the entire procedure can be regarded as a hard attention mechanism. Similar with this idea, hard monotonic attention [16] with end-to-end differentiable method was introduced so that the hard attention can be better trained with expectation instead of sampling approximation in [14, 15]. However, these hard attention mechanisms access a single memory item for each decoding step which is much worse than exploiting the whole input memory with soft attention. Thus, monotonic chunkwise attention (MoChA) [17] was proposed to allow the model to perform soft attention over small chunks of the input memory preceding where a hard monotonic attention mechanism has chosen to attend. Benefiting from the small chunks, MoChA yields comparable online performance with conventional soft attention. However, MoChA uses a fixed chunk size for all output symbols which neglects the variational durations of output symbols.

The online attention mechanisms mentioned above indeed work, but the segmental structure of speech is underutilized more or less. There are work [18, 19] to regard the segmentation of input speech sequence as a latent random variable, and the sequence transduction is performed with all alignment paths marginalized out via dynamic programing. However, these models either introduce conditional independence assumptions between output symbols or require a fixed alignment for training. Meanwhile, online speech recognition is not explored in their work.

Therefore, here comes the question: How to utilize the segmental structure of speech for online speech recognition? To this end, we propose a segment boundary detection directed attention mechanism which splits the input speech into successive segments with detected boundaries so that different output symbols adaptively have different chunk sizes for aggregating information within the segment with soft attention. The segment boundaries are detected by a boundary detector which is trained jointly with the encoder-decoder model. By explicitly introducing a boundary detector to utilize the segmental structure, we can easily model the monotonic alignment between input and output sequence and make the attention-based encoder-decoder model applicable to online speech recognition. In our method, segment boundary detection directs the online recognition in a strictly monotonic manner; thus, it is required that an output symbol should be corresponding to a segment of speech. And unpronounced output symbols are not suitable for our method, such as the often-used character in English where some letters are not pronounced in words. More discussions can be found in Section 4.2.2.

The rest of this paper is structured as follows: in Section 2, we first give a brief discussion about hard attention and segmental patterns. Then, we introduce two models most relevant to our work for comparison: conventional attention based encoder-decoder model and hard alignment with RL. Afterwards, we describe our proposed approach including training and inference procedure in Section 3. Experiments, discussion, and conclusion are given in the following sections.

2 Related work

2.1 Hard attention and segmental pattern

As a kind of attention mechanisms, hard attention selects specific inputs and discards others for decoder to integrate information from input sequence. These discrete operations often require RL style approaches to train [2]. Meanwhile, from RL perspective, hard attention can be formulated as an agent that interacts with an input sequence over time and learns good policies to take discrete actions about where to pay attention in the sequence. There are previous works that use RL to learn where to consume the input and/or when to emit an output [20] for tasks like image classification [21], object detection [22], object tracking [23], and speech recognition [14]. Similarly, in our work, we formulate the segment boundaries as actions which is suitable for online speech recognition.

Not only in speech, segmental patterns exist in many types of sequences such as phrases in human languages [24]. Therefore, it is quite natural to consider how to incorporate the segmental structure into sequence-to-sequence models. Many works try to model the various valid sequence segmentations by introducing a latent variable [18, 19, 25, 26]. Although following the same idea, the segment boundaries of speech can be regarded as a latent variable; we notice that it is equivalent to our formulation of regarding the boundaries as actions when introducing some approximations (Jensen's inequality) for the latent variable. More details and derivations can be found in [2, 27].

2.2 Conventional attention-based encoder-decoder model

As mentioned in Section 1, the conventional attention-based encoder-decoder model contains three parts: encoder to process input sequence $X = (x_1, x_2, \ldots, x_T)$, decoder to produce output sequence $Y = (y_1, y_2, \ldots, y_N)$, and attention mechanism to align the sequence X and Y. The encoder converts input sequence to a memory bank $H = (h_1, h_2, \dots, h_T)$. And the decoder attends input memory H and generates i_{th} output symbol y_i based on the attended context representation c_i, the decoder state s_i, and the previous output y_{i-1}. When applying encoder-decoder model to ASR, the input X is usually a sequence of feature vectors like filter bank features, and the output Y is a sequence of phonemes, characters, or words. The forward pass of the model is given by the following equations:

$$H = \text{Encoder}(X) \tag{1}$$
$$\tilde{s}_i = \text{RNN}(s_{i-1}, y_{i-1}) \tag{2}$$
$$\alpha_i = \text{Attend}(\tilde{s}_i, \alpha_{i-1}, H) \tag{3}$$
$$c_i = \sum_{k=1}^{T} \alpha_i^k h_k \tag{4}$$
$$s_i = \text{RNN}(\tilde{s}_i, c_i) \tag{5}$$
$$y_i = \text{softmax}(W[\,s_i; c_i; y_{i-1}]\,) \tag{6}$$

where s_i is the decoder state at output step i and $\tilde{s}_i$ is the intermediate variable. The attention weight vector is $\alpha_i \in \mathbb{R}^T$, known as the alignment of input memory and output sequence. The Attend() function in Eq. (3) is often modeled by a multiple layer perceptron (MLP) equipped with convolutional feature shown below:

$$e_i^k = \text{MLP}(\tilde{s}_i, h_k, [F * \alpha_{i-1}]^k\,) \tag{7}$$
$$\alpha_i^k = e_i^k \Big/ \sum_{k=1}^{T} e_i^k \tag{8}$$

Note that the attention weight α_i is calculated based on the entire input memory H, which is a problem for online speech recognition.

2.3 Hard online alignment with policy gradient

To address the online problem, Luo [14] introduced hard online alignment to get an online sequence-to-sequence model. The online model uses binary stochastic variables to select the timesteps at which to produce outputs. The recognition process can be described as follows: at each input timestep t, a recurrent neural network with hidden state h_t decides whether or not to emit an output symbol. The decision is represented by a stochastic binary logistic unit b_t. Let $\tilde{b}_t \sim \text{Bernoulli}(b_t)$ be a Bernoulli distribution so that if $\tilde{b}_t$ is 1, then the model emits an output symbol. During training, the binary decision is generated by sampling from Bernoulli distribution, and during inference, the decision is set to 1 if the binary distribution probability b_t exceeds a certain threshold. The current decoding position in the output sequence Y can be written as $\tilde{p}_t = \sum_{j=1}^{t} \tilde{b}_j$, which is incremented by 1 every time the model chooses to emit. Then, the model's goal is to predict the desired output $\tilde{y}_i = y_{\tilde{p}_t}$. At each input timestep t, the binary decision from the previous timestep $\tilde{b}_{t-1}$ and the previous target $\tilde{y}_{i-1}$ are fed into the model as input. They used reinforcement learning to train the network on when to emit the various outputs and supervised learning to train the network to make the correct output predictions. The reward of the model is defined in Eq. (15).

The main recognition process of the model is summarized by the following equations:

$$h_t = \text{RNN}(h_{t-1}, [\,x_t; \tilde{b}_{t-1}; \tilde{y}_{i-1}]\,) \tag{9}$$
$$b_t = \text{sigmoid}(W_b h_t) \tag{10}$$
$$\tilde{b}_t \sim \text{Bernoulli}(b_t) \tag{11}$$
$$\tilde{p}_t = \sum_{j=1}^{t} \tilde{b}_j \tag{12}$$
$$\tilde{y}_i = y_{\tilde{p}_t} \tag{13}$$
$$d_t = \text{softmax}(W_o h_t) \tag{14}$$
$$R = R + \tilde{b}_t \cdot \log d_t(\tilde{y}_i) \tag{15}$$

By factoring the reward R into $R(\tilde{\mathbf{b}})$ and distribution $p(\tilde{\mathbf{b}})$ over the binary decision sequence $\tilde{\mathbf{b}}$, the loss is:

$$L = E_{\tilde{\mathbf{b}}}[\,R(\tilde{\mathbf{b}})] \tag{16}$$

where the expectation E is based on $\tilde{\mathbf{b}} = (\tilde{b}_1, \tilde{b}_2, \dots, \tilde{b}_T)$.

And the gradient estimate with respect to the parameters is:

$$\nabla L = E_{\tilde{\mathbf{b}}}[\,\nabla R(\tilde{\mathbf{b}}) + R(\tilde{\mathbf{b}}) \nabla \log p(\tilde{\mathbf{b}})] \tag{17}$$

The gradient consists of two parts: Given a value of the binary decision $\tilde{b}_t$, the gradient of the model can be calculated with backpropagation (BP) algorithm by maximizing the output likelihood (the first term in the right side of Eq. (17)). The gradient of the binary random variable $\tilde{b}_t$ in Eq. (11) can be calculated with policy gradient (PG) (the second term in the right side of Eq. (17)). The whole model is a stochastic computation graph that includes both deterministic functions (output sequence Y) and conditional probability distributions (binary decision sequence $\tilde{\mathbf{b}}$). The detailed derivations can be found in [20, 28].

While the hard online alignment enables the real-time processing of the speech stream, there exists an issue for the model to predict output by attending only a single input memory item. To remedy this issue, we propose a segment boundary detection directed attention which allows the decoder to attend input memory within detected segments. Although motivated by Luo's work introduced above, our method incorporates the segmental structure into the attention mechanism which is more reasonable for online alignment, and as a consequence, the online performance is significantly improved compared to Luo's work (see Table 3).

3 Method

3.1 Encoder-decoder model with segment boundary detection directed attention

The diagram of the online encoder-decoder model with our proposed attention mechanism is shown in Fig. 1. In the diagram, both encoder and attention are different from the conventional attention model: the encoder is a unidirectional recurrent network; the soft attention is performed over a segment with boundary indicator $\tilde{b}_t$ decided by a recurrent boundary detector. These differences, suitable for online recognition, enable the decoder to produce output symbols based on the detected segments of the input speech stream.

Given an input sequence $X = (x_1, x_2, \ldots, x_T)$, firstly, the recurrent encoder converts the current input x_t to hidden state h_t. Then, the recurrent boundary detector updates its hidden state S^a_{t-1} with $\{h_t, \tilde{s}_i, \tilde{b}_{t-1}\}$ as input, which represents encoder state h_t, decoder intermediate state $\tilde{s}_i$ ($\tilde{s}_i$ is the query of attention mechanism), and binary indicator $\tilde{b}_{t-1}$ about whether or not the previous input x_{t-1} is a segment boundary, respectively. After the update, the detector predicts the boundary probability b_t of current input with new state S^a_t and makes a decision $\tilde{b}_t$ based on probability b_t (Eqs. (18–22)).

Note that, similar with Luo's work in Section 2.3, the binary decision $\tilde{b}_t$ is generated by sampling during training (Eq. (22)) and by threshold decision during inference. The formula of the boundary probability in Eq. (21) is the same as [16], allowing the model to learn appropriate offset r for the pre-sigmoid activation and make the probability nonsensitive to the scale of the activation. We initialize r to a negative value -2 before training to prevent the sampled boundaries from occurring in the very beginning of the input sequence.

$$h_t = \text{Encoder}(h_{t-1}, x_t) \tag{18}$$

$$\tilde{s}_i = \text{RNN}(s_{i-1}, y_{i-1}) \tag{19}$$

$$S^a_t = \text{RNN}(S^a_{t-1}, [\,h_t; \tilde{s}_i; \tilde{b}_{t-1}]\,) \tag{20}$$

$$b_t = \text{sigmoid}(g\frac{v^\top}{\|v\|}\tanh\left(W_b S^a_t\right) + r) \tag{21}$$

$$\tilde{b}_t \sim \text{Bernoulli}(b_t) \tag{22}$$

As shown in Fig. 1, the output production is triggered by the input boundary detection. If the decision $\tilde{b}_t$ is 0 (shaded round node) which means current input is not a segment boundary, the model continues to access next input x_{t+1} without emitting an output symbol. Otherwise,

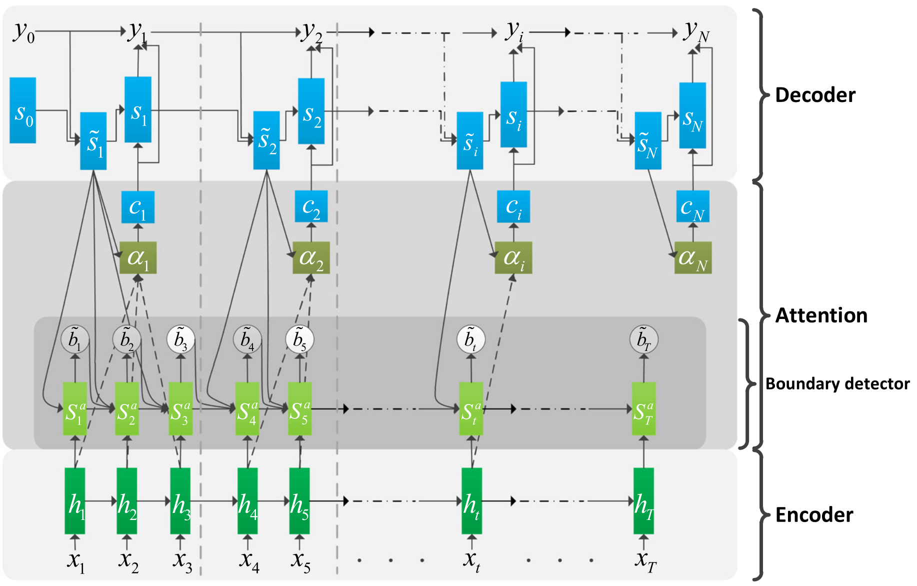

Fig. 1 Diagram of the online encoder-decoder model with segment boundary detection directed attention. The input and output sequence are $(x_1, x_2, \ldots, x_T)$ and $(y_1, y_2, \ldots, y_N)$, respectively. $\tilde{b}_t$ denotes the random variable of binary segment boundary decision, α_i is attention weight vector, c_i is the attended context vector—weighted summation of the input hidden states, and h_t, S^a_t, s_i, $\tilde{s}_i$ are the hidden states of the recurrent networks. y_0 is the StartOfSequence symbol and s_0 is the initial decoder state. The encoder processes the input stream frame-by-frame, and boundary decisions are made for each input. A shaded $\tilde{b}_t$ indicates that current input is not a segment boundary for the model and the decoder stays idle while an unshaded $\tilde{b}_t$ mean that current input is a segment boundary and the decoder should produce an output by attending the corresponding segment with soft attention. For instance, the decisions for the first three inputs are $\tilde{\mathbf{b}}_{1:3} = (0, 0, 1)$ which means input x_3 is a detected segment boundary. Then, the decoder predicts y_1 by attending the hidden states of the segment (h_1, h_2, h_3). After that, a new input x_4 is received and this procedure continues until the end of the input or output sequence is reached

if the decision $\tilde{b}_t$ is 1 (unshaded round node) which means current input is a segment boundary, the model attends the corresponding segment with range $[z_{i-1}+1, z_i]$ and emits an output y_i:

$$z_i = t \quad \text{if}\ (\tilde{b}_t = 1) \tag{23}$$

$$e_i^k = \text{MLP}(\tilde{s}_i, h_k, [F * \alpha_{i-1}]^k) \tag{24}$$

$$\alpha_i^k = \frac{e_i^k}{\sum_{t=z_{i-1}+1}^{z_i} e_i^t} \tag{25}$$

$$c_i = \sum_{t=z_{i-1}+1}^{z_i} \alpha_i^t h_t \tag{26}$$

$$s_i = \text{RNN}(\tilde{s}_i, c_i) \quad \text{if}\ (\tilde{b}_t = 1) \tag{27}$$

$$p(y_i) = \text{softmax}(W[s_i; c_i; y_{i-1}]) \tag{28}$$

$$i = 1 + \sum_{j=1}^{t} \tilde{b}_j \tag{29}$$

The index t means current input timestep and the index i means current output step (both t and i start from 1). The input timestep t always updates once new input x_t is received while the output step i updates only when $\tilde{b}_t$ is 1 (Eq. (29)).

The main difference with Luo's work in Section 2.3 is shown in Eqs. (23–26). When a segment boundary is detected, the segment range is also determined by previous boundary z_{i-1} and current boundary z_i. Based on that, the decoder performs soft attention over the i_{th} detected segment $(h_{z_{i-1}+1}, h_{z_{i-1}+2}, \ldots, h_{z_i})$ and predicts the corresponding output symbol y_i with decoder state $\tilde{s}_i$, encoder context vector c_i and previous output y_{i-1}. This procedure continues until the end of the input or output sequence is reached.

In order to get better performance with the proposed attention mechanism, we refine its actual implementation based on the architecture described above. The refinements are based on two considerations: (1) phones can be better recognized if more context acoustic information is included. Therefore, when a segment boundary is detected, the soft attention range can be extended by combining the corresponding segment with its restricted left and right context. Meanwhile, this extension makes the model more robust to the possible boundary detection errors. In our experiments, we extend the left context by concatenating previous and current detected segment ("Extend left" in Table 1). As for the right context, we extend the segment with 2 following input memory items for low latency recognition ("Extend right" in Table 1). Specifically, for both "Extend left" and "Extend right," the attention range $[z_{i-1}+1, z_i]$ in Eqs. (25–26) is replaced with $[z_{i-2}+1, z_i+2]$. (2) The acoustic variation of consecutive input frames is essential for boundary detection [29]. Thus, both past and future frame information should be provided to the boundary detector when inspecting each potential boundary frame. The past frame information can be utilized by the recurrent boundary detector. As for the future information, inspired by the unidirectional long short-term memory (LSTM) [8] with target delay [30] to access some future context, we introduce a decision delay ("Decision delay" in Table 1) which allows the model to look ahead some future input memory items when making a boundary decision. Specifically, for a boundary decision sequence with 2 delayed timesteps $(N_{ull}, N_{ull}, \tilde{b}_1, \tilde{b}_2, \ldots, \tilde{b}_t)$, the input sequence of the boundary detector is $(h_1, h_2, \ldots, h_t, h_{t+1}, h_{t+2})$.

Table 1 Test PER results of segment boundary detection directed attention model with refinements on TIMIT dataset

Extend left	Extend right	Decision delay	PER (%)
×	×	0	26.3
✓	×	0	25.0
✓	✓	0	24.1
✓	✓	1	23.5
✓	✓	2	22.7
✓	✓	4	22.5

3.2 Training and inference procedure

The online process described above involves sampling, which is beyond the reach of standard BP algorithm. Therefore, as discussed in Section 2.1, we formulate the segment boundary detection as a sequential decision-making problem and solve it with RL algorithm. The boundary detector is a goal-oriented *agent* which interacts with speech and takes *actions* about whether or not current input is a segment boundary. The *state* of the agent contains four elements: the current input hidden state h_t, the previous action $\tilde{b}_{t-1}$, the current decoder state $\tilde{s}_i$, and the interaction history $Z_{1:t} = \{(h_1, \tilde{s}_1), \tilde{b}_1, \ldots, (h_{t-1}, \tilde{s}_{i'}), \tilde{b}_{t-1}\}$ (a sequence of past observations and actions, $i' \leq i$). In our model, the state of the agent is summarized in the RNN hidden state S_t^a (Eq. 20). Similar with Luo's work in Section 2.3, the *reward* is defined as:

$$r_t = \begin{Bmatrix} \log p(y_i), & \text{if}\ (\tilde{b}_t = 1) \\ 0, & \text{else} \end{Bmatrix} \tag{30}$$

If the action $\tilde{b}_t$ is 1, the reward is the log likelihood of current output symbol. Otherwise, the reward is 0.

With these RL-related terms specified, the boundary detector can be optimized by maximizing the total reward it can expect in the long run:

$$L = E_{\tilde{\mathbf{b}}}[R(\tilde{\mathbf{b}})] \tag{31}$$

where $\tilde{\mathbf{b}} = (\tilde{b}_1, \tilde{b}_2, \ldots, \tilde{b}_T)$ and R is the accumulated reward of the sequence: $R = \sum_{t=1}^{T} r_t$. The loss here is the same with Eq. (16) introduced in Section 2.3; thus, the gradient estimate in Eq. (17) can be reused. The gradient of the binary random variable $\tilde{b}_t$ (second term in the right side of the Eq. (17)) is shown below:

$$E_{\tilde{\mathbf{b}}}[R(\tilde{\mathbf{b}})\nabla\log\rho(\tilde{\mathbf{b}})] \approx \frac{1}{M}\sum_{i=1}^{M}\sum_{t=1}^{T}\nabla\log p\left(\tilde{b}_t^i|S_{t,i}^a\right)\left(R_t^i-\mu_t\right) \tag{32}$$

where the expectation E is approximated with sampling and M is the number of sampled sequences. R_t^i is the accumulated reward of the i_{th} sequence from time t to T: $R_t^i = \sum_{l=t}^{T} r_l^i$. μ_t is a baseline which is introduced to lower the variance of the gradient approximation. Similar to works [21, 23], we select $\mu_t = E_{\tilde{\mathbf{b}}}[R_t] \approx \frac{1}{M}\sum_{i=1}^{M} R_t^i$ as our baseline.

In order to successfully train the model, we adopt two tricks used in [14]: (1) the sampled binary decisions in Eq. (22) is modified by forcing $\tilde{b}_t$ to be equal to 1 whenever $T - t \leq N - i$ to ensure that the model is forced to generate the entire output sequence during training. T and N are the input and output sequence length, respectively. (2) An entropy penalty for boundary probability b_t is used to prevent too confident binary decisions and improve the exploration of the agent. Following [14, 31], the entropy penalty can be incorporated by defining an augmented reward:

$$r_t^{\text{aug}} = r_t - \lambda\left(\tilde{b}_t\log(b_t) + (1-\tilde{b}_t)\log(1-b_t)\right) \tag{33}$$

where λ is the weight of entropy penalty. And our model is trained with this augmented reward.

During inference, we use a modified beam-search algorithm presented in Algorithm 1 to generate output sequences. The key of the modified beam-search algorithm is to synchronize the boundary detection timesteps of each beam so that all beams can emit output symbols from different start positions. The decoding procedure for each beam is similar to the online process in Section 3.1, with a difference that the boundary decision in Eq. (22) is replaced with $\tilde{b}_t = \mathbb{I}(b_t > \tau)$ where $\mathbb{I}$ is the indicator function and τ is a threshold. In the decoding procedure, both the boundary probability assignment and the output symbol production rely on the interaction histories including past inputs, outputs, and actions. Therefore, at each decoding step, the detected boundaries in each beam may be different, which results in the output symbols getting emitted at different input timesteps. A maximum allowable decoding delay D (Line 11 in Algorithm 1) is used to prevent the model from consuming all encoder frames without emitting output and make the model applicable in online recognition. In order to get fair and comparable beam scores for each beam, beams with already detected segment boundaries should wait for other beams and the beam pruning is conducted until every beam has encountered a segment boundary and emitted an output (Line 24 in Algorithm 1). The beam pruning and candidate generation are the same with conventional beam-search algorithm. And the latest detected boundary location z_i is stored in a beam candidate together with the output subsequence Y_i. After the new beams are determinated, the decoding process continues with the start position of the input boundary detection probably different for each beam. And the decoding process is terminated either all beams emit EndOfSequence symbol or reach the end of the input sequence.

4 Experimental results and discussion

We validate our model on two datasets: a small one called TIMIT [32] and a large one called WSJ [33]. Initial experiments are conducted on TIMIT to assess the refinements described in Section 3.1 and the online performance of the proposed attention mechanism. Based on that, the behavior of the model including the segment boundary detection output and the learned online alignment are analyzed on TIMIT to understand how the attention mechanism works. Besides, experiments and segment boundary detection output on WSJ dataset are presented to provide more convincing results.

4.1 Experiments with small dataset: TIMIT

4.1.1 Dataset and model setup

The first set of experiments are conducted on TIMIT which is a widely used speech corpus for phoneme recognition task. In our experiments, we use the standard data partition of train/valid/test set which contains 3696, 400, and 192 utterances respectively. The input to encoder is 40-dimensional mel scale filter bank features together with the energy in each frame, and their first and second temporal differences, yielding in total 123-dimensional features per frame. And the output is 61 phone set with an extra "EndOfSequence" token. Decoding is performed with beam-search using the 61+1 phone set, while scoring is conducted on the standard 39 phone set. Phone error rate (PER) is used as the evaluation criterion.

In order to compare with recent end-to-end methods for online recognition [14–17], we use a unidirectional encoder consisting of 3 gated recurrent unit (GRU) [9] layers with 512 hidden units and a single unidirectional GRU decoder with 256 hidden units. And the hidden state sequence of the final encoder layer is downsampled by every three hidden states. This results in a 1/3 downsampling rate so that the sequence processed by the segment boundary detector is shorter, which allows the model to learn proper boundary decisions much easier with RL. Convolutional feature (see Eq. (3)) is used in our models with same configuration as [3]. A single layer GRU with 256 hidden units is adopted as the

Algorithm 1 Beam-search algorithm for segment boundary detection directed attention model.

```
1:  Input: $H$: input memory bank with length $T$, $N$: maximum decoding step, $V$: output vocabulary, $K$: beam width, $D$: maximum allowable decoding delay
2:  Output: $F$: set of completed output sequences
3:  Declaration:
    $B$: set of beam candidates, $Score$: scores of each beam
    $SeqsNew$: set of decoded output subsequence and latest detected boundary location pairs $(Y_i, z_i)$ at current output step
    $SeqsOld$: set of decoded output subsequence and latest detected boundary location pairs $(Y_{i-1}, z_{i-1})$ at previous output step
4:  Initialization: $F \Leftarrow \{\emptyset\}$, $Score(\{\emptyset\}) = 0$, $SeqsOld = \{(y_0, z_0)\}$, $S_0^a = \vec{0}$, $b_0 = 0$, $z_0 = 0$, $i = 1$, $y_0 =$ StartOfSequence
5:  while $i \le N$ do    // Produce output symbols until the decoded output subsequence length larger than N
6:    $SeqsNew \Leftarrow \{\emptyset\}$, $B \Leftarrow \{\emptyset\}$
7:    for subsequence and detected boundary location pair $(Y_{i-1}, z_{i-1})$ in $SeqsOld$ do
8:      while $z_{i-1} + 1 \le j \le T$ do    // Start inspecting memory items $h_j$s left-to-right from where we left off
9:        $S_j^a = \mathrm{RNN}(S_{j-1}^a, [\,h_j; \tilde{s}_i; \tilde{b}_{j-1}\,])$    // Update the agent state
10:       $b_j = \mathrm{sigmoid}(g \frac{v^\top}{\|v\|} \tanh(W_b S_j^a) + r)$    // Compute segment boundary probability of current input
11:       if $b_j \ge \tau$ or $j - z_{i-1} \ge D$ then    // If $b_j$ is larger than threshold $\tau$ or delay is larger than D, we stop scanning the memory
12:         $z_i = j$    // Set current segment boundary location
13:         $c_i = \sum_{t=z_{i-1}+1}^{z_i} \alpha_i^t h_t$    // Segment-directed soft attention with $\alpha_i$ computed based on Eq. (25)
14:         $p(y_i|Y_{i-1}; h_{1\sim z_i}) = \mathrm{softmax}(W[\,s_i; c_i; y_{i-1}\,])$    // Compute output probability of symbol $y_i$
15:         for $y_i$ in vocabulary $V$ do    // Extend the beam candidate with all possible output in V
16:           $Y_i \Leftarrow \mathrm{concate}(Y_{i-1}, y_i)$
17:           $Score((Y_i, z_i)) = Score((Y_{i-1}, z_{i-1})) + \log p(y_i|Y_{i-1}; h_{1\sim z_i})$
18:           Add $(Y_i, z_i)$ to the set of beam candidates $B$
19:         end for
20:         break    // Stop searching memory items for this beam in SeqsOld
21:       end if
22:     end while    // Break the loop when an output is emitted or the memory sequence is over
23:   end for    // Finish inspecting memory items for all beams in SeqsOld
24:   $SeqsNew \Leftarrow$ The top-$K$ beams $(Y_i, z_i)$ from $B$ in terms of $Score$
25:   for $(Y_i, z_i)$ in $SeqsNew$ do    // Filter out subsequences who generated EndOfSequence symbol
26:     if $y_i =$ EndOfSequence then    // The last symbol of the subsequence $Y_i$ is $y_i$
27:       Add $Y_i$ to $F$
28:       Delete tuple $(Y_i, z_i)$ in $SeqsNew$
29:       $K = K - 1$
30:     end if
31:   end for
32:   $SeqsOld \Leftarrow SeqsNew$
33:   if $SeqsNew = \{\emptyset\}$ then
34:     break
35:   end if
36: end while
```

recurrent boundary detector because of GRU's capability of memorizing long historical data. The weight of the entropy penalty λ described in Section 3.2 is set to 1 in the beginning and linearly decays from 1 to 0.3 during 5000 to 10000 iteration steps. Then, the weight is set to 0.3 after 10000 steps. In our experiments, we use GRU layer to replace LSTM layer since GRU has comparable performance to LSTM [34] while GRU consumes fewer parameters and is more efficient to be trained.

We use Adadelta optimizer [35] with $\epsilon = 1e-8$ and $\rho = 0.95$, and the model is trained with dropout rate 0.3 [36, 37] for first 10 epochs. After lowest validation negative log-likelihood is achieved, we continue to train the model with adaptive weight noise [38] and lower the ϵ to

$1e-9$ if we do not observe the performance improvement for successive 10 epochs in validation PER. Finally, we stop training if there is no more gain in the development set. Beam-search is used during inference with beam size 10, and the threshold of the boundary probability τ is set to 0.35. Maximum allowable decoding delay D is set to 20. The minibatch size is 5 and sample size M is 15. The global gradient clipping is set to 10. As model performance varies with different initial weights, we conduct 5 repeat trials for each model and report the average PER. All the models for TIMIT experiments are implemented based on Theano [39].

4.1.2 Model performance

We first investigate the effect of the refinements introduced in Section 3.1 and further choose a proper model architecture for following experiments. Table 1 shows the results of models with refinements including "Extend left", "Extend right" and "Decision delay." Different from other experiments on TIMIT, all the models here are trained without adaptive weight noise [38] for simplicity. The PER of the model without extensions is 26.3%. By extending the soft attention area with left and right context, a lower PER 24.1% is obtained. And the PER is reduced as the decision delay increases. Considering the demand of low latency, we adopt decision delay with 2 input timesteps in the following TIMIT experiments.

Therefore, the segment boundary detection directed attention model with all the refinements is denoted as "Proposed method" in Table 2, and all the models implemented here are trained with adaptive weight noise. To further verify the effectiveness of our model and exclude the impact of extra introduced parameters (GRU boundary detector), we conduct more experiments for three baseline models equipped with the conventional offline soft attention: encoder-decoder model with same encoder and decoder size to our proposed model (denoted as "soft attention"), soft attention model with one more unidirectional GRU layer with 512 hidden units stacked on the encoder (denoted as "soft attention bigger-E"), and soft attention model with decoder replaced by 2 GRU layers with 320 hidden units (denoted as "soft attention bigger-D"). These baseline models do not have any downsampling layers. We also list the online performance of previous end-to-end models in Table 2. And the number of parameters of previous models in the table are estimated based on the literatures. It is shown that our proposed model yields PER 20.2% which is comparable with the best online encoder-decoder models. And the model with GRU boundary detector is more effective than the baseline models with more GRU layers in encoder or decoder.

The learning curves of various models are shown in Fig. 2. Although our model is trained with RL whose approximated gradient has high variance, the training progress on TIMIT is steady and the convergence speed is similar with soft attention (boundary detection directed attention vs. soft attentions). As for WSJ, it converges slower than baseline models. We hypothesize that utterances in TIMIT are shorter which makes it easier for the model to explore various boundary trials and receive instructive feedbacks.

We also show the development and evaluation set accuracies of different decoding thresholds in Fig. 3 to evaluate how fragile the model is towards the threshold selection. The threshold value varies from 0.25 to 0.55 on TIMIT and 0.2 to 0.5 on WSJ with increment 0.05. As we can see, the performance remains stable in a narrow range, like 0.35–0.45 on TIMIT and 0.25–0.35 on WSJ, which means we should be careful about the threshold. And the maximum delay D reduces the decoding errors when a large decision threshold is used. D is tuned based on the development set and a smaller D may degrade the performance because of hurting the long silence skipping ability of the model.

Table 2 Test PER results of different models for online recognition on TIMIT dataset

Model	#Param (M)	PER (%)
Partial condition [12]	3.1	20.8
Hard alignment with RL [14]	6.8	20.5
Gaussian prediction attention [40]	5.8	20.4
Hard monotonic attention [16]	6.4	20.4
Stacked LSTM [15]	1.0	20.0
CTC [41]	3.8	19.6
Proposed method	6.9	20.2
Soft attention*	5.9	21.0
Soft attention bigger-E*	7.5	20.8
Soft attention bigger-D*	7.0	20.6

All the models use a unidirectional encoder and * indicates offline attention model

4.1.3 Case study

In addition to the recognition performance, we also conduct a case study to analyze how the proposed attention mechanism works by examining the segment boundary detection output, segment-directed soft attention, and recognition result of an example from validation set. The boundary detection output shown in Fig. 4 is generated with recognized hypothesis **HYP** as decoder input, which is consistent with inference procedure. The speech spectrogram is visualized in the first row of the figure, together with ground-truth phone segments represented by dash lines and reference transcription denoted in each segment. From the spectrogram, we can easily find most of the phone segment boundaries based on the acoustic

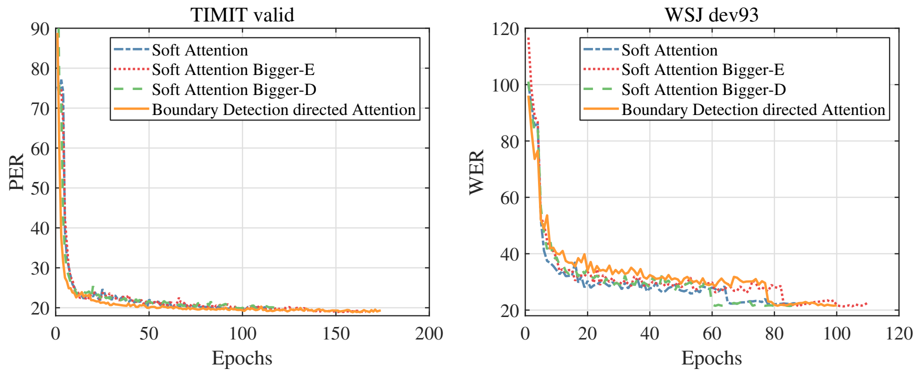

Fig. 2 Learning curves of several attention models. Left: The valid PER for each epoch on TIMIT dataset. Right: The valid WER for each epoch on WSJ dataset with dev93 as validation set. The "soft attention," also denoted as baseline model, means a model equipped with conventional soft offline attention while using the same unidirectional encoder and decoder of our proposed model. And the "soft attention bigger-E" means the baseline model with another GRU layer stacked on the encoder. "Soft attention bigger-D" means the baseline model with another GRU layer stacked on the decoder. Both "soft attention bigger" models have similar or more amount of parameters with our proposed model (see Table 2). The "boundary detection directed attention" means our proposed model in this work. The sudden drops in dev93 WER represent the learning rate decay during training procedure

variations of the consecutive frames. And this makes the segment boundary detection possible. In order to compare the detected boundaries with the spectrogram, we set the x-ticks differently so that the length of the boundary probabilities and decisions are the same with the spectrogram (the downsampling rate is 1/3 in our model). And the recognized hypothesis is also denoted in each detected segment. As is shown in the figure, the detected boundaries (red bars in the second row of Fig. 4) are roughly precise in the front part of the utterance and become disordered in the ending. Correspondingly, there are 7 recognition errors (1 insertion error, 2 deletion errors, 4 substitution errors) in this example, and the detailed reference transcript and recognized hypothesis are given below:

REF: *h# w ay bcl* *ay q oy l w* ***ax*** *n y* ***ux*** *q* ***ao w ey*** *z y ux z m ay* ***n*** *h#*

HYP: *h# w ay bcl* ***b*** *ay q oy l w* ***ix*** *n y* ***uw*** *q* ***oy*** *z y ux z m ay* ***m*** *h#*

As we can see, many phones are correctly recognized given the roughly precise segment boundaries. However, deletion error may occur (***w ey*** in **REF**) if a real boundary is missed, and insertion error may occur (***b*** in **HYP**) if a boundary is falsely detected within the real segment. These errors suggest that more precise boundary detection methods are needed.

In order to investigate how the boundary detection affects the alignment, we visualize the segment boundary detection directed attention and the conventional soft offline attention in Fig. 5. Note that reference **REF** is used as decoder input and the encoder of the baseline model is downsampled with 1/3 rate so as to make the two attentions comparable. And the ground-truth phone segments (dark dash line) and detected segments (red dash line) are given in the figure with reference transcription attached. As the online decoding goes on, the maximum score of the segment-directed attention may occur at the detected boundary, within the detected segment or in

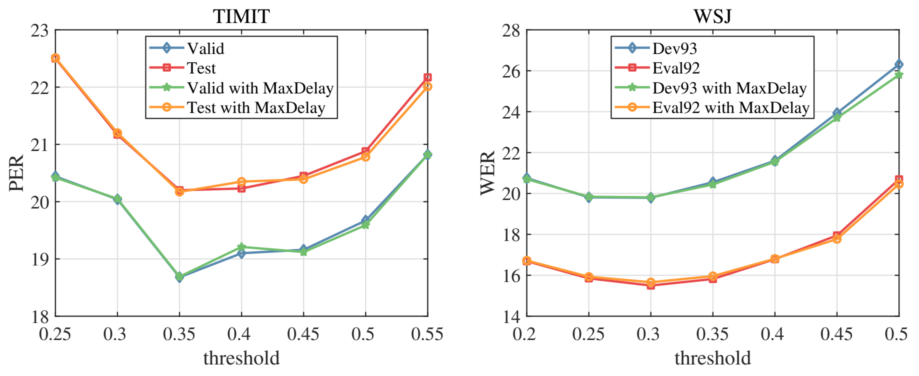

Fig. 3 Development and evaluation set accuracies with different decoding thresholds

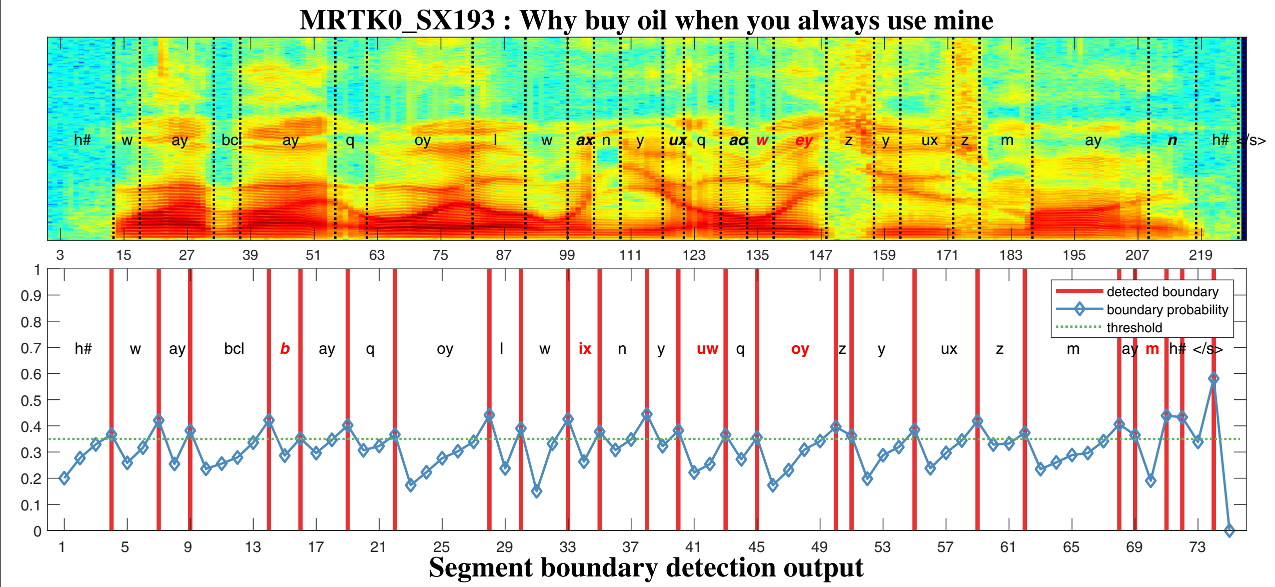

Fig. 4 Speech spectrogram and segment boundary detection output for utterance MRTK0_SX193 from validation set. Top: Speech spectrogram with ground-truth phone segments represented by dash lines and reference transcription denoted in each segment. Bottom: Segment boundary detection output generated with recognized hypothesis as decoder input. The blue line is the boundary probability of each input memory item ranging from 0 to 1 (most of the probabilities are around 0.2 to 0.6). The green dashed line indicates the threshold to emit output symbols which is set to 0.35 on TIMIT. And the red bars are detected boundaries based on threshold decision. The frame rate of boundary sequence is 1/3 of original input speech because of 1/3 downsampling rate in our model. And recognized hypothesis is also denoted in each detected segment with recognition errors marked in red

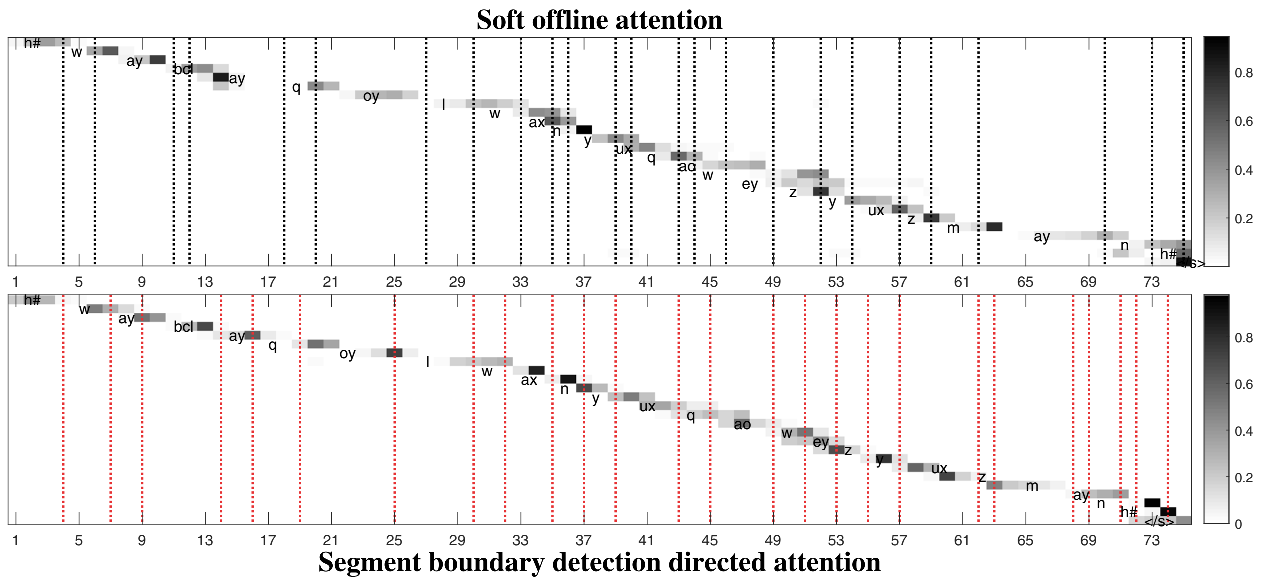

Fig. 5 Conventional offline attention and segment boundary detection directed attention for utterance MRTK0_SX193 from validation set. Reference transcription is used as decoder input for both models and each phone symbol is denoted to its corresponding alignment. The soft offline attention is generated by the "soft attention bigger-D" model, which contains a downsampling layer with 1/3 downsampling rate in encoder and consumes comparable amounts of parameters to boundary detection directed attention model (see Table 2). The dark dash lines in the first row indicate ground-truth phone segments with 1/3 downsampling rate. The red dash lines in the second row indicate the detected segment boundaries

the extended area ("Extend left/right"). And the segment-directed attention is similar to the offline attention even though some boundary detection errors are made. We hypothesize that the model may benefit from the decoder output history and the extended attending range, which makes the model robust to boundary detection errors.

4.2 Experiments with large dataset: WSJ

4.2.1 Dataset and model setup

Limited by the corpus size, the performance of encoder-decoder models on TIMIT is often highly affected by the hyperparameters and regularization tricks. Considering that, we also validate our model on WSJ dataset to make the experimental results more convincing. In WSJ, 81 h long SI-284 set is used for training, "dev93" set is used for validation, and "eval92" set is used for evaluation. As we failed to reproduce the powerful baseline model used in [16, 17], the baseline model proposed in [16, 17] is modified by adding dense connections [42–44] between layers of the encoder and using subword output unit, in order to get a new baseline model in our work with comparable performance. Dense connection is adopted because it facilitates the training of deep models.

The details of our baseline encoder-decoder model are described below. The input to deep convolutional encoder is 80-dimensional mel scale filter bank features together with their first and second temporal differences, which is organized as a $T \times 80 \times 3$ tensor with 80 as frequency dimension and static, first, and second temporal differences stacked as 3 feature maps. This input sequence is first fed into two convolutional layers, each with 3×3 filters and a 2×2 stride with 64 feature maps per layer. Each convolution is followed by batch normalization (BN) [45] prior to a rectified linear unit (ReLU) activation. After processed by the two convolutional layers with 2×2 stride, the input sequence has 1/4 downsampling rate and is passed into a convolutional LSTM layer with 64 feature maps, using 1×3 filters. This was followed by an additional 3×3 convolutional layer with 64 feature maps and a 1×1 stride. These two convolutional layers are considered as a dense block thus each layer has dense connections to its posterior layers. Transition layer with 256 hidden units is introduced and there is no bottleneck layer within the block. Then, the encoder had another dense block consisting of three additional unidirectional GRU layers with 256 hidden units, each followed by a dense layer with 256-dimensional output, BN and ReLU activation. And dense connections are added to each GRU output. Dropout is embedded after every ReLU activation. The decoder was a single unidirectional GRU layer with 256 hidden units and takes a 64-dimensional learned embedding as input. The attention module of the baseline is a windowing approach with width 200 and convolutional feature dimension 50, a little different from [4]. The softmax output layer took as input the concatenation of decoder's state, attended context vector, and embedding of previous output token (see Eqs. (6, 28)).

The mostly used output unit for WSJ is character while subword units have proven to be better than characters [46]. Another consideration is that subword may be more relevant to phonetic unit and thus suitable for segment boundary detection. More discussion can be found in Sections 4.2.2 and 4.2.3. Therefore, we employ byte-pair encoding (BPE) as output unit [47], and 1K BPE units are used in our experiments, similar to [48]. The beam-search decoding will go over these BPE units, and at the end of decoding, the BPE units are merged into words in order to obtain the best hypothesis on word level. And external language model is not used in our experiments.

As for our proposed attention mechanism, same with the refinements in TIMIT experiments, we extend the attending range for each detected segment in both directions so as to make the model more robust to segment boundary detection errors. And decision delay is not used since the encoder contains 2 convolutional layers which have already encoded the information of several future frames to the hidden representations of each potential boundary frame.

We use Adam optimizer [49] with learning rate $2.5e-4$ for segment boundary detection directed attention model and dropout is active after 10 epochs with rate 0.5. The learning rate is set to $2.5e-5$ and $2.5e-6$ respectively if no performance improvement is observed in validation word error rate (WER) for successive 10 epochs. The entropy penalty weight λ is set to 1 for first 30 epochs and linearly decays to 0.3 for next 20 epochs and then keeps 0.3 for the remaining epochs. The minibatch size is 8 and the sample size M is 4. The global gradient clipping is set to 1. The threshold τ of segment boundary probability is set to 0.3 and maximum allowable decoding delay D is set to 40. As for the baseline models with conventional offline soft attention, the learning rate is set to $5e-4$, and same decay scheme is used with factor 10. Baseline models are trained with minibatch size 30. Both the baseline and our proposed model are trained with label smoothing 0.1 from the beginning [50]. Four GPU are employed with synchronous updating to speed up the training. All the models for WSJ experiments are implemented based on Tensorflow [51]. For each model, we conducted 5 repeat trials and report the mean and standard deviation of WER.

4.2.2 Model performance

The experimental results are listed in Table 3. We estimated the number of parameters of previous models based on literatures and list them in the table for comparison. Our offline soft attention baseline has 5.4M parameters and yields 15.3% WER with deviation of 0.3% while the baseline reported in [17] is 14.6% $\pm$ 0.3%. We have

Table 3 Test WER results of different models for online recognition on WSJ dataset

Model	#Param (M)	WER (%)
Hard alignment with RL [14]	2.7	27.0
CTC [52]	–	22.7
Hard monotonic attention [16]	2.9	17.4
MoChA [17]	3.0	15.0±0.6
Soft attention* [17]	2.9	14.6±0.3
Proposed method (BPE)	6.6	15.5±0.4
Soft attention (BPE)*	5.4	15.3±0.3
Soft attention bigger-E (BPE)*	6.7	15.1±0.3
Soft attention bigger-D (BPE)*	6.7	15.0±0.3
Proposed method (Char)	6.6	22.2±0.5
Soft attention bigger-D (Char)*	6.7	16.3±0.3

All the models use a unidirectional encoder and * indicates offline attention model

tried several encoder architectures and found that the model described above yields the lowest WER. Thus, we conducted an online recognition with our proposed attention mechanism and the modified encoder and obtained comparable performance with state-of-the-art MoChA model (15.5% vs. 15.0%). Although our model has 6.6M parameters, two times larger than MoChA which has 3.0M parameters, we think it is mainly the baseline soft attention models rather than attention mechanisms that occupy most of the parameters. Therefore, without loss of generality, we compared our method with our own soft attention models and treated MoChA as a reference. Similar to experiments on TIMIT, we also stack one more unidirectional GRU layer to the encoder/decoder of the baseline model (denoted as bigger-E/bigger-D) which consumes similar amount of parameters with our proposed model (6.7M vs. 6.6M). And the result shows the performance of the proposed online attention mechanism is close to the offline soft attention. Based on that, we can confirm that the segment boundary detection directed attention is effective for online speech recognition. Meanwhile, we also aware of that the proposed attention mechanism is not easy to be well trained with PG algorithm since segment boundary is explicitly detected with hard decisions. The learning curve (BPE as output unit) in Fig. 2 shows that the segment boundary detection directed attention converges a little slower than soft attention but is steady on WSJ although it is trained with PG. Therefore, improving the training efficiency of the model is necessary in future work.

Moreover, we ran additional experiments with character as output unit for better comparison with BPE unit. All configurations are the same with BPE-based models except that the threshold τ is set to 0.5. In Table 3, the performance of proposed method degrades drastically from 15.5 to 22.2% and the gap between soft attention and proposed method (16.3% vs. 22.2%) is very large for character. Comparing with MoChA where character is used as output, our character-based model performs worse. These results validate our viewpoint that character is not suitable for our method as explained in the introduction. In MoChA and offline soft attention, input speech frames or chunks are not forced to uniquely map to an output. For example, in MoChA, multiple outputs can be emitted for a chunk, because the new inspection of memory entries starts from where it left off (see Algorithm 1 in [17]). Based on that, unpronounced labels could probably be emitted without consuming new input frames. However, in our method, each detected segment can only emit a single output (line 8 and 12 in Algorithm 1), which requires each output label to correspond with a speech segment. Therefore, unpronounced output characters would confuse the segment boundary detector. And long words may also compel the detector to trigger the output character emission frequently, which increases the burden of boundary detection and results in shorter segment length. In contrast, BPE is more relevant to phonetic unit than character and thus more suitable for segment boundary detection directed attention. We also provide a case study to better illustrate this in Fig. 7. We believe that a minimum segment length and allowing overlapping segment detections are worth exploring in future work to handle the higher output frame rate for character-based output.

4.2.3 *Case study*

In order to further analyze the behavior of the boundary detector, the segment boundary detection output (BPE) of a longer WSJ dev utterance with long silence parts is shown in Fig. 6. Similar to TIMIT example in Section 4.1.3, the boundary detection output shown in Fig. 6 is generated with recognized BPE hypothesis **HYP** as decoder input, which is consistent with inference procedure. And all detected boundaries of BPE symbols are represented by dark dashed lines. Since manually annotated boundaries are unavailable on WSJ, we obtain "ground-truth" word boundaries from a HMM-GMM model with Kaldi [53]. The speech spectrogram is visualized in the first row of the figure, together with HMM-GMM-generated word segments represented by dash lines and reference transcription denoted in each segment. Silence symbol "sil" is automatically inserted to the original reference **REF**. In the second row of the figure, we only denote recognized word sequence in each detected word segment (red lines) for simplicity and word boundary is considered as the detected boundary of a word's last piece.

As is shown in the figure, a BPE symbol is usually emitted when new speech stream reaches. However, the model is not robust to long silence part, for example,

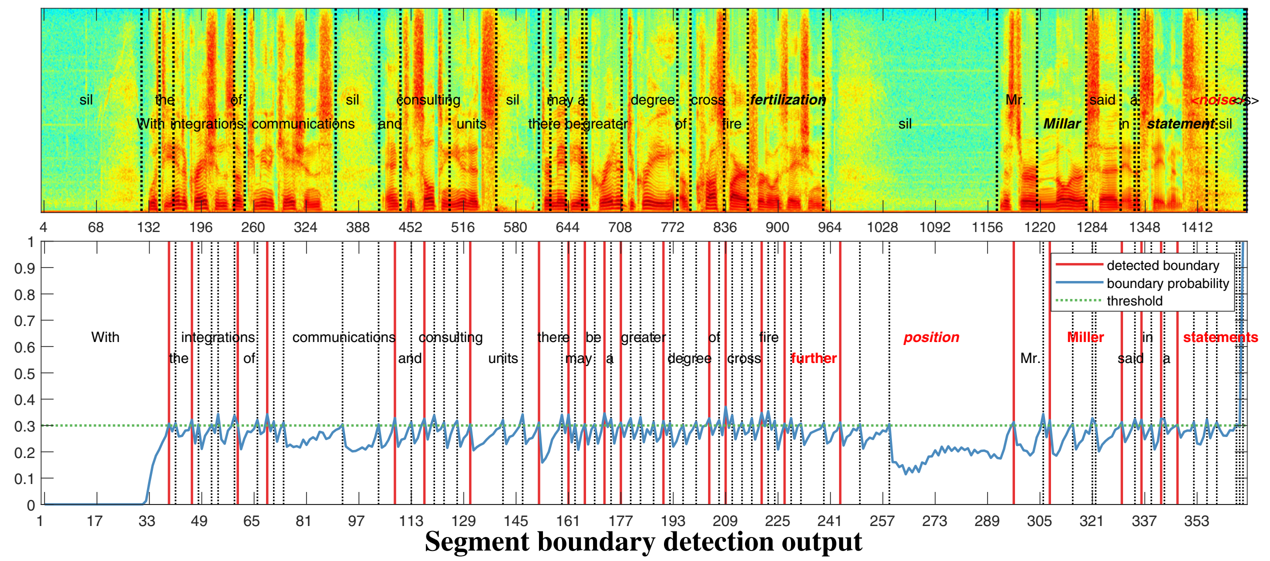

Fig. 6 Speech spectrogram and segment boundary detection output (BPE) for utterance 4k0c0303 from "dev93" set. Top: Speech spectrogram with word segments (forced alignment generated by a HMM-GMM model) represented by dash lines and reference transcription denoted in each segment. Bottom: Segment boundary detection output generated with recognized hypothesis as decoder input. The blue line is the boundary probability of each input memory item ranging from 0 to 1 (most of the probabilities are around 0.1 to 0.4). The green dashed line indicates the threshold to emit output symbols which is set to 0.3 on WSJ. The dark dashed lines are detected boundaries of BPE outputs based on threshold decision. And the red lines are the detected boundaries of last piece of a word, which can be considered as word boundaries. The frame rate of boundary sequence is 1/4 of original input speech because of 1/4 downsampling rate in our model. And recognized hypothesis is also denoted in each detected segment with recognition errors marked in red

"fertilization" followed by long silence is falsely recognized to "further position." The detailed reference transcript and recognized hypothesis as well as BPE sequence **BPE_HYP** are given below:

REF: *With the integrations of communications and consulting units there may be a greater degree of cross fire* ***fertilization*** *Mr.* ***Millar*** *said in a* ***statement*** *<* ***noise*** *>*

HYP: *With the integrations of communications and consulting units there may be a greater degree of cross fire* ***further position*** *Mr.* ***Miller*** *said in a* ***statements***

BPE_HYP: *With < spc> the < spc> in te gr ations < spc> of < spc> comm un ic ations < spc> and < spc> con sul ting < spc> un its < spc> there < spc> may < spc> be < spc> a < spc> gre at er < spc> de g ree < spc> of < spc> c ro ss < spc> fir e < spc> f ur ther < spc> posi tion < spc> Mr. < spc> M ill er < spc> said < spc> in < spc> a < spc> st at em ents*
"< spc>" symbol represents space between words and the BPE sequence is corresponding to detected boundaries denoted as dark dashed lines and red lines in Fig. 6.

In order to complement the discussion about the character-based model in Section 4.2.2, we present another example with utterance including unpronounced characters. The BPE-based and character-based segment boundary detection outputs are shown in Fig. 7. Note that word "knowing" is "k n o w i n g" for character-based model and is "kno w ing" for BPE-based model. For character-based model, word "knowing" with unpronounced "k" is falsely recognized to "no in", while for BPE-based model, "knowing" is correctly recognized after merging subwords "kno w ing". And BPE boundaries are more compatible with acoustic variations.

Although we have not conducted experiments for much larger dataset like LibriSpeech (960 h) [54] due to limited computing resources and time, both recognition accuracies and case studies have shown the effectiveness of our model for online speech recognition. And to progress, experiments with larger datasets as well as other performance improving techniques including decoding with language model is necessary in future work.

5 Conclusion

In this paper, we present a new online attention mechanism: the segment boundary detection directed attention which divides the conventional soft attention into two parts—segment boundary detection and segment-directed soft-attention. By formulating the boundary detection as a sequential decision-making problem, the online attention mechanism can be trained with RL algorithm so that online end-to-end speech recognition is achieved. Experiments on TIMIT and WSJ dataset have demonstrated the effectiveness of our proposed model whose performance is comparable with the conventional offline soft attention models and state-of-the-art online

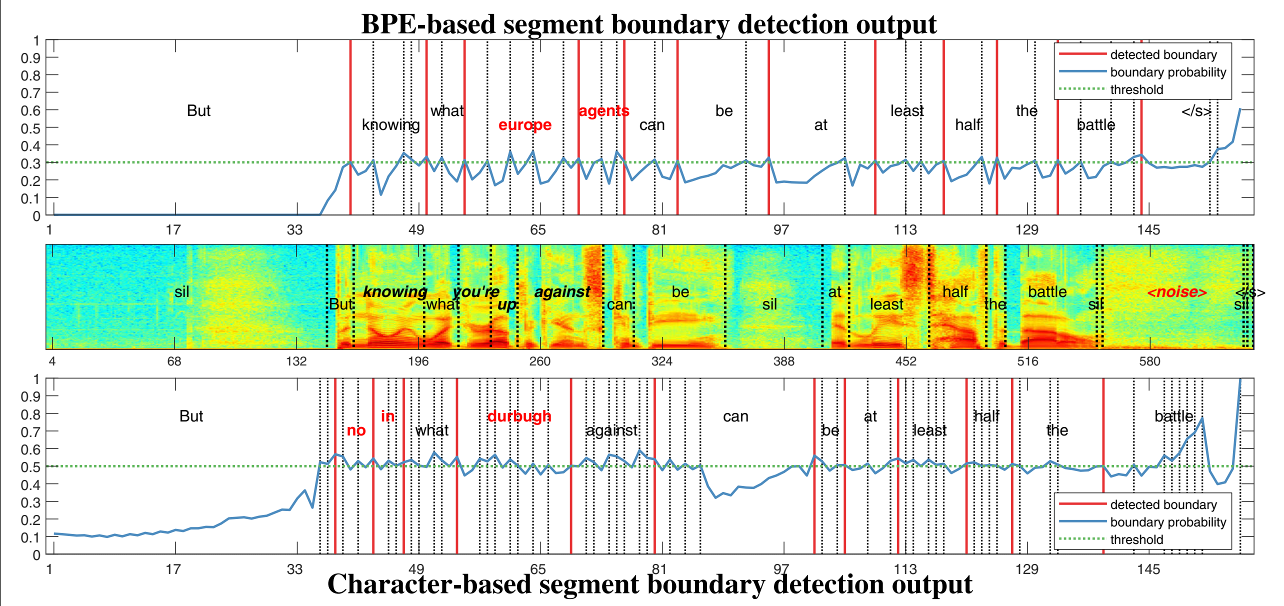

Fig. 7 Speech spectrogram and BPE-based/character-based segment boundary detection output for utterance 4k0c030j from "dev93" set. Top: BPE-based segment boundary detection output. Middle: Speech spectrogram with word segments (forced alignment generated by a HMM-GMM model) represented by dash lines and reference transcription denoted in each segment. Bottom: Character-based segment boundary detection output. Both the detection outputs are generated with recognized hypothesis as decoder input. The blue lines are the boundary probabilities of each input memory item. The green dashed lines indicate the threshold to emit output symbols which is set to 0.3 for BPE and 0.5 for character. The dark dashed lines are detected boundaries of BPE/character outputs based on threshold decision. And the red lines are the detected boundaries of last piece/character of a word, which can be considered as word boundaries. The frame rate of boundary sequence is 1/4 of original input speech because of 1/4 downsampling rate in our model. And recognized hypotheses are also denoted in each detected segment with recognition errors marked in red

attention models. Although some detection errors exist, the online alignment learned by the proposed attention mechanism is roughly similar to soft offline alignment, which can explain the effectiveness of the mechanism. Both the recognition performance and the segment boundary detection output have provided evidence that utilizing segmental structure of speech for online speech recognition is reasonable as well as effective. In the future, we will explore more boundary detection methods to improve the training efficiency and the boundary detection accuracy so as to conduct experiments on much larger datasets and further improve the online recognition performance.

Abbreviations

ASR: Automatic speech recognition; BN: Batch normalization; BP: Backpropagation; BPE: Byte-pair encoding; GRU: Gated recurrent unit; LSTM: Long short-term memory; MLP: Multiple layer perceptron; MoChA: Monotonic chunkwise attention; PER: Phone error rate; PG: Policy gradient; ReLU: Rectified linear unit; RL: Reinforcement learning; RNN: Recurrent neural network; WER: Word error rate

Acknowledgements

We would like to thank the editor and anonymous reviewers for their careful work and thoughtful suggestions that help to improve this paper substantially.

Funding

This work was supported by National Key R&D Program of China (Grant No. 2017YFB1002202), the National Natural Science Foundation of China (Grant No. U1836219) and the Key Science and Technology Project of Anhui Province (Grant No. 18030901016).

Authors' contributions

JH and LD conceived the algorithm. JH wrote the software, executed the experiments, and drafted the document. WG and YS provided helpful advice and support during the time of designing the algorithm and experiments. LD supervised the research. JH and LD completed the final manuscript. All authors read and approved the final manuscript.

Availability of data and materials

The datasets used and analyzed during the current study are available from the Linguistic Data Consortium. TIMIT dataset is available from (https://catalog.ldc.upenn.edu/LDC93S1). WSJ dataset is available from (https://catalog.ldc.upenn.edu/LDC93S6B, https://catalog.ldc.upenn.edu/LDC94S13B).

Competing interests

The authors declare that they have no competing interests.

Received: 7 April 2019 Accepted: 13 January 2020
Published online: 30 January 2020

References

1. D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate. arXiv preprint (2014). arXiv:1409.0473
2. K. Xu, J. L. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. S. Zemel, Y. Bengio, in *32nd International Conference on Machine Learning, ICML 2015*. Show, attend and tell: Neural image caption generation with visual attention, vol. 3 (International Machine Learning Society (IMLS), 2015), pp. 2048–2057. https://nyuscholars.nyu.edu/en/publications/show-attend-and-tell-neural-imagecaption-generation-with-visual-

3. J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, Y. Bengio, in *NIPSŠ15: Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*. Attention-based models for speech recognition (MIT Press, Cambridge, 2015), pp. 577–585. https://dl.acm.org/doi/proceedings/10.5555/2969239
4. D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, Y. Bengio, in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. End-to-end attention-based large vocabulary speech recognition, (2016), pp. 4945–4949. https://doi.org/10.1109/icassp.2016.7472618
5. W. Chan, N. Jaitly, Q. Le, O. Vinyals, in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Listen, attend and spell: a neural network for large vocabulary conversational speech recognition, (2016), pp. 4960–4964. https://doi.org/10.1109/icassp.2016.7472621
6. S. Watanabe, T. Hori, S. Kim, J. R. Hershey, T. Hayashi, Hybrid CTC/attention architecture for end-to-end speech recognition. IEEE J. Sel. Top. Signal Process. **11**(8), 1240–1253 (2017)
7. S. Hochreiter, J. Schmidhuber, Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
8. H. Sak, A. Senior, F. Beaufays, in *INTERSPEECH-2014*. Long short-term memory recurrent neural network architectures for large scale acoustic modeling, (2014), pp. 338–342. https://www.isca-speech.org/archive/interspeech_2014/i14_0338.html
9. K. Cho, van Merrienboer Bart, G. Caglar, B. Dzmitry, B. Fethi, S. Holger, B. Yoshua, in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Learning phrase representations using RNN encoder–decoder for statistical machine translation, (2014), pp. 1724–1734. https://doi.org/10.3115/v1/d14-1179
10. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*. Attention is All you Need, (2017), pp. 5998–6008. https://dblp.uni-trier.de/rec/bibtex/conf/nips/VaswaniSPUJGKP17
11. W. Chan, I. Lane, in *Proc. Interspeech 2016*. On online attention-based speech recognition and joint Mandarin character-Pinyin training, (2016), pp. 3404–3408. https://doi.org/10.21437/interspeech.2016-334
12. N. Jaitly, Q. V. Le, O. Vinyals, I. Sutskever, D. Sussillo, S. Bengio, in *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*. An Online Sequence-to-Sequence Model Using Partial Conditioning, (2016), pp. 5067–5075. https://dblp.uni-trier.de/rec/bibtex/conf/nips/JaitlyLVSSB16
13. N. Moritz, T. Hori, J. L. Roux, in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Triggered attention for end-to-end speech recognition, (2019), pp. 5666–5670. https://doi.org/10.1109/icassp.2019.8683510
14. Y. Luo, C. Chiu, N. Jaitly, I. Sutskever, in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Learning online alignments with continuous rewards policy gradient, (2017), pp. 2801–2805. https://doi.org/10.1109/icassp.2017.7952667
15. C. Chiu, D. Lawson, Y. Luo, G. Tucker, K. Swersky, I. Sutskever, N. Jaitly, An online sequence-to-sequence model for noisy speech recognition. arXiv preprint (2017). arXiv:1706.06428
16. C. Raffel, T. Luong, P. J. Liu, R. J. Weiss, D. Eck, Online and linear-time attention by enforcing monotonic alignments. arXiv preprint (2017). arXiv:1704.00784
17. C. Chiu, C. Raffel, Monotonic chunkwise attention. arXiv preprint (2017). arXiv:1712.05382
18. L. Lu, L. Kong, C. Dyer, N. A. Smith, S. Renals, in *Proc. Interspeech 2016*. Segmental recurrent neural networks for end-to-end speech recognition, (2016), pp. 385–389. https://doi.org/10.21437/interspeech.2016-40
19. E. Beck, M. Hannemann, P. Dötsch, R. Schlüter, H. Ney, in *Proc. Interspeech 2018*. Segmental encoder-decoder models for large vocabulary automatic speech recognition, (2018), pp. 766–770. https://doi.org/10.21437/interspeech.2018-1212
20. W. Zaremba, I. Sutskever, Reinforcement learning neural turing machines-revised. arXiv preprint (2015). arXiv:1505.00521
21. V. Mnih, N. Heess, A. Graves, K. Kavukcuoglu, in *Proceedings of the 27 th International Conference on Neural Information Processing Systems - Volume 2*. Recurrent Models of Visual Attention (MIT Press, Cambridge, 2014), pp. 2204–2212. https://dl.acm.org/doi/10.5555/2969033.2969073
22. S. Mathe, A. Pirinen, C. Sminchisescu, in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Reinforcement learning for visual object detection, (2016), pp. 2894–2902. https://doi.org/10.1109/cvpr.2016.316
23. D. Zhang, H. Maei, X. Wang, Y.-F. Wang, Deep reinforcement learning for visual object tracking in videos. arXiv preprint (2017). arXiv:1701.08936
24. C. Wang, Y. Wang, P.-S. Huang, A. Mohamed, D. Zhou, L. Deng, in *Proceedings of the 34th International Conference on Machine Learning - Volume 70*. Sequence modeling via segmentations, (2017), pp. 3674–3683
25. L. Yu, J. Buys, P. Blunsom, in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online segment to segment neural transduction, (2016), pp. 1307–1316. https://doi.org/10.18653/v1/d16-1138
26. L. Kong, C. Dyer, N. A. Smith, Segmental recurrent neural networks. arXiv preprint (2015). arXiv:1511.06018
27. D. Lawson, C. Chiu, G. Tucker, C. Raffel, K. Swersky, N. Jaitly, in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Learning hard alignments with variational inference, (2018), pp. 5799–5803. https://doi.org/10.1109/icassp.2018.8461977
28. J. Schulman, N. Heess, T. Weber, P. Abbeel, in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*. Gradient Estimation Using Stochastic Computation Graphs (MIT Press, Cambridge, 2015), pp. 3528–3536. https://dl.acm.org/doi/10.5555/2969442.2969633
29. Y.-H. Wang, C.-T. Chung, H.-Y. Lee, in *Proc. Interspeech 2017*. Gate activation signal analysis for gated recurrent neural networks and its correlation with phoneme boundaries, (2017), pp. 3822–3826. https://doi.org/10.21437/interspeech.2017-877
30. A. Graves, J. Schmidhuber, Framewise phoneme classification with bidirectional lstm and other neural network architectures. Neural. Netw. Off. J. Int. Neural. Netw. Soc. **18**, 602–10 (2005)
31. R. J. Williams, J. Peng, Function optimization using connectionist reinforcement learning algorithms. Connect. Sci. **3**(3), 241–268 (1991)
32. J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, D. S. Pallett, DARPA TIMIT acoustic-phonetic continous speech corpus CD-ROM. NIST speech disc 1-1.1. NASA STI/Recon technical report n. 93 (1993)
33. D. B. Paul, J. M. Baker, in *Proceedings of the Workshop on Speech and Natural Language*. The design for the Wall Street Journal-based CSR corpus, (1992), pp. 357–362. https://doi.org/10.6028/nist.ir.4930
34. J. Chung, Ç. Gülçehre, K. Cho, Y. Bengio, Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint (2014). arXiv:1412.3555
35. M. D. Zeiler, Adadelta: an adaptive learning rate method. arXiv preprint (2012). arXiv:1212.5701
36. N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**, 1929–1958 (2014)
37. W. Zaremba, I. Sutskever, O. Vinyals, Recurrent neural network regularization. arXiv preprint (2014). arXiv:1409.2329
38. A. Graves, in *Proceedings of the 24th International Conference on Neural Information Processing Systems*. Practical Variational Inference for Neural Networks (Curran Associates Inc., Red Hook, 2011), pp. 2348–2356. https://dl.acm.org/doi/10.5555/2986459.2986721
39. Theano Development Team, Theano: A Python framework for fast computation of mathematical expressions. arXiv preprint (2016). arXiv:1605.02688
40. J. Hou, S. Zhang, L.-R. Dai, in *Proc. Interspeech 2017*. Gaussian prediction based attention for online end-to-end speech recognition, (2017), pp. 3692–3696. https://doi.org/10.21437/interspeech.2017-751
41. A. Graves, A. Mohamed, G. Hinton, in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Speech recognition with deep recurrent neural networks, (2013), pp. 6645–6649. https://doi.org/10.1109/icassp.2013.6638947
42. G. Huang, Z. Liu, L. v. d. Maaten, K. Q. Weinberger, in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Densely connected convolutional networks, (2017), pp. 2261–2269. https://doi.org/10.1109/cvpr.2017.243
43. C. Y. Li, N. T. Vu, Densely connected convolutional networks for speech recognition. arXiv preprint (2018). arXiv:1808.03570
44. Z. Ding, R. Xia, J. Yu, X. Li, J. Yang, Densely connected bidirectional lstm with applications to sentence classification. arXiv preprint arXiv:1802.00889 (2018)

45. S. Ioffe, C. Szegedy, in *Proceedings of the 32nd International Conference on Machine Learning*. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift (PMLR, Lille, 2015), pp. 448–456. http://proceedings.mlr.press/v37/ioffe15.html
46. C. Chiu, T. N. Sainath, Y. Wu, R. Prabhavalkar, P. Nguyen, Z. Chen, A. Kannan, R. J. Weiss, K. Rao, E. Gonina, N. Jaitly, B. Li, J. Chorowski, M. Bacchiani, in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. State-of-the-Art Speech Recognition with Sequence-to-Sequence Models, (2018), pp. 4774–4778. https://ieeexplore.ieee.org/abstract/document/8462105
47. R. Sennrich, B. Haddow, A. Birch, in *ACL*. Neural machine translation of rare words with subword units, (2016), pp. 1715–1725
48. A. Zeyer, K. Irie, R. Schlüter, H. Ney, Improved training of end-to-end attention models for speech recognition. arXiv preprint (2018). arXiv:1805.03294
49. D. P. Kingma, J. Ba, in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Adam: A Method for Stochastic Optimization, (2015). https://dblp.org/rec/bibtex/journals/corr/KingmaB14
50. G. Pereyra, G. Tucker, J. Chorowski, L. Kaiser, G. E. Hinton, Regularizing neural networks by penalizing confident output distributions. arXiv preprint (2017). arXiv:1701.06548
51. M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, X. Zheng, TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems (2015). https://github.com/tensorflow/docs/blob/master/site/en/about/bib.md
52. C. Wang, D. Yogatama, A. Coates, T. X. Han, A. Y. Hannun, B. Xiao, in *Workshop Extended Abstracts of the 4th International Conference on Learning Representations*. Lookahead Convolution Layer for Unidirectional Recurrent Neural Networks, (2016). https://www.semanticscholar.org/paper/Lookahead-Convolution-Layer-for-Unidirectional-Wang-Yogatama/a0d864d73189101a0bffc6656aa907f3b2193cfa
53. D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, K. Vesely, in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. The Kaldi Speech Recognition Toolkit (IEEE Signal Processing Society, 2011). http://kaldi-asr.org/doc/about.html
54. V. Panayotov, G. Chen, D. Povey, S. Khudanpur, in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Librispeech: an ASR corpus based on public domain audio books, (2015), pp. 5206–5210. https://doi.org/10.1109/icassp.2015.7178964

Publisher's Note