Listen Attentively, and Spell Once: Whole Sentence Generation via a Non-Autoregressive Architecture for Low-Latency Speech Recognition

Ye Bai\textsuperscript{1,2}, Jiangyan Yi\textsuperscript{1}, Jianhua Tao\textsuperscript{1,2,3}, Zhengkun Tian\textsuperscript{1,2}, Zhengqi Wen\textsuperscript{1}, Shuai Zhang\textsuperscript{1,2}

\textsuperscript{1}NLPR, Institute of Automation, Chinese Academy of Sciences, Beijing
\textsuperscript{2}School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing
\textsuperscript{3}CAS Center for Excellence in Brain Science and Intelligence Technology
\{ye.bai, jiangyan.yi, jhtao, zhengkun.tian, zqwen, shuai.zhang\}@nlpr.ia.ac.cn

Abstract

Although attention based end-to-end models have achieved promising performance in speech recognition, the multi-pass forward computation in beam-search increases inference time cost, which limits their practical applications. To address this issue, we propose a non-autoregressive end-to-end speech recognition system called LASO (listen attentively, and spell once). Because of the non-autoregressive property, LASO predicts a textual token in the sequence without the dependence on other tokens. Without beam-search, the one-pass propagation much reduces inference time cost of LASO. And because the model is based on the attention based feedforward structure, the computation can be implemented in parallel efficiently. We conduct experiments on publicly available Chinese dataset AISHELL-1. LASO achieves a character error rate of 6.4\%, which outperforms the state-of-the-art autoregressive transformer model (6.7\%). The average inference latency is 21 ms, which is 1/50 of the autoregressive transformer model.

Index Terms: speech recognition, sequence-to-sequence, non-autoregressive, transformer

1. Introduction

Attention based sequence-to-sequence (Seq2Seq) speech recognition systems have achieved promising performance these years \cite{1, 2, 3}. In these models, an encoder encodes acoustic features into high-level representations. And a decoder is a conditional language model, which predicts the next token in the sequence. For an autoregressive model, the prediction of one token relies on the previously predicted tokens at inference stage. The non-autoregressive Seq2Seq models assume that each token is independent from the others:

\begin{equation}
    P(y|x) = P(y_1|x) \prod_{i=2}^{L} P(y_i|y_{<i}, x),
\end{equation}

where \(y_i\) denotes the token at step \(i\), \(y_{<i}\) denotes the subsequence \([y_1, \ldots, y_{i-1}]\), and \(L\) denotes the length of token sequence. For an autoregressive model, the prediction of one token relies on the previously predicted tokens at inference stage:

\begin{equation}
    P(y|x) = \prod_{i=1}^{L} P(y_i|x).
\end{equation}

Because the prediction does not depend on other tokens, the non-autoregressive Seq2Seq model can predict the token at each step in parallel.

2. Background

Speech recognition aims to convert an acoustic feature sequence to the corresponding textual token (word, sub-word, or phone) sequence. Given a speech-text pair \((x, y)\), where \(x\) denotes the acoustic feature sequence, and \(y\) denotes the token sequence, the autoregressive Seq2Seq model estimates the conditional probability \(P(y|x)\) by decomposition with the chain rule:

We believe that the language semantic\textsuperscript{1} is contained in the speech signal implicitly. So, if this semantic can be extracted well, the token sequence can be generated without relying on the explicit language modeling, e.g., autoregressive language models and masked language models. In this paper, we propose a simple and effective non-autoregressive model called LASO (Listen Attentively, and Spell Once\textsuperscript{2}). We use the feed-forward self attention mechanism \cite{4} as basic blocks to build three modules of LASO: the encoder, the position dependent summarizer (PDS), and the decoder. The encoder encodes the acoustic features into high-level representations. The PDS summarizes the semantic at each position from the high-level representations and bridges length gap between speech and token sequence. The decoder captures token-level semantic and predicts tokens. We conduct experiments on a publicly available Chinese dataset AISHELL-1\textsuperscript{3}. The proposed LASO achieves 6.4\% of character error rates on test set, which is better than chain model \cite{12} and state-of-the-art autoregressive transformer models \cite{13}. And compared with the strong baseline autoregressive transformer model, the inference of LASO speeds up by 50\times.

\textsuperscript{1}In this paper, we refer to the relationship among tokens as language semantic.

\textsuperscript{2}This name is inspired by \cite{3}.

\textsuperscript{3}This name is inspired by \cite{3}.
Token relationship is important for sequence generation. For the CTC based models [14], the token relationship is usually modeled by an external language model to improve performance. For RNN-Transducers [15], the token relationship is modeled with a prediction network. And for attention based encoder-decoder models, the token relationship is modeled with the decoder autoregressively. The main challenge of the non-autoregressive Seq2Seq model is: can a model generate token sequence without the explicit token relationship? We believe the probability distribution on vocabulary at each position. The tail of the token sequence is filled by "<eos>". The network is trained with cross entropy.

Token relationship is important for sequence generation. For the CTC based models [14], the token relationship is usually modeled by an external language model to improve performance. For RNN-Transducers [15], the token relationship is modeled with a prediction network. And for attention based encoder-decoder models, the token relationship is modeled with the decoder autoregressively. The main challenge of the non-autoregressive Seq2Seq model is: can a model generate token sequence without the explicit token relationship? We believe that the token relationship is contained in the speech implicitly. If we achieve token-level representations from speech, we can generate the token sequence with the non-autoregressive model.

3. The LASO Model
The basic idea of LASO is that the acoustic feature sequence contains not only features for pronunciation but also language semantic, i.e., relationship among tokens. If we extract representations from the whole acoustic feature sequence for each token position, we can do position-wise token prediction. Because the prediction relies on the acoustic feature sequence rather than other tokens, it can be implemented in parallel.

Based on this idea, we formulate the position-wise token prediction as

\[ z = \text{Encode}(x), \]
\[ P(y_i|x) = \text{SummarizeAndDecode}(z), i = 1, 2, \ldots, L \]

where \( z = [z_1, \ldots, z_T] \) is the hidden representation sequence which has the same length with the subsampled acoustic feature sequence \( x \). To predict the token sequence with length \( L \), \( z \) is summarized and decoded for each position in the sequence. To generate the token sequence, the most likely token at each position is selected. The token "<eos>" is added into the vocabulary as a filler for padding the token sequence to length \( L \). Ideally, the tail of the generated token sequence are all "<eos>", and they are easily removed after inference.

The proposed LASO consists of three modules. Each module consists of several attention blocks. The encoder encodes the acoustic feature sequence into high-level representations. The PDS summarizes the high-level representations based on the sinusoidal position encodings. The decoder generates the token for each position. The structure of the model is shown in Fig. 1. We first introduce the attention block. Then, we introduce each module of the model.

3.1. Attention Block
Attention mechanism has been used to model global dependency in a sequence successfully [4, 16]. Different from recurrent neural networks which represent context step-by-step, attention mechanism fuses all representations in a sequence by weighting sum. So, it can be computed in parallel. The dot-product self-attention is denoted as

\[ \text{Attention}(Q, K, V) = \text{Softmax}(QK^TW) \]
The position-wise feedforward network is after the attention:
\[
    \text{FFN}(x) = W_{2}\text{Activation}(W_{1}x + b_{1}) + b_{2},
\]
where \( x \) is a vector at one position, \( W_{1}, W_{2}, b_{1}, \) and \( b_{2} \) are learnable parameters, Activation is a nonlinear activation function. In this work, we use gated linear units (GLUs) [17]. Residual connection [18] and layer normalization [19] are used in the attention block. We use pre-norm mechanism for stable training [20]. The attention block is the basic component of LASO.

3.2. Encoder
The first part of the encoder consists of two layers of convolutional neural network (CNN) for capturing locality of in the feature sequence. The stride of each CNN layer is 2, so it also subsamples frame rates and compress the length of the sequence to 1/4. Following [4], we add sinusoidal position for self attention mechanism to capture the order. Then, \( N_{e} \) attention blocks are used for capturing long-term relationship. Keys, queries, and values are all the inputs, i.e., self attention.

3.3. Position Dependent Summarizer
The main gap between the acoustic feature sequence and the textual token sequence is the length. Specifically, a textual token is a highly compressed semantic representation, and softmax functions values and queries are the outputs of the previous block. After \( 3.4 \), the queries of the first block are position encodings from the encoder outputs. So, the sequence length matches the codings provide position dependent information to query representation from the encoder, and to re-organize them in terms of the token positions. Basically, it is also composed of a stack of attention blocks, but the keys and the values are the outputs of the encoder. The queries of the first block are position encodings with maximum length \( L \), and the queries of the follow-up blocks are the outputs of the previous block, as shown in Fig. 1. We use sinusoidal functions [4] to encode positions:
\[
\begin{align*}
    p_{c,e;x}, & \text{, } z_{j} = \sin(i/10000^{2j/D_{m}}), \quad \text{ \( i = 1, \ldots, L \) denotes the i-th position}, \\
    p_{c,v;x}, & \text{, } z_{j+1} = \cos(i/10000^{2j/D_{m}}),
\end{align*}
\]
where \( i = 1, \ldots, L \) denotes the \( i \)-th position, and \( 2j \) and \( 2j+1 \) denote element indexes in a vector. The sinusoidal position encodings provide position dependent information to query representations corresponding to specific position in token sequence from the encoder outputs. So, the sequence length matches the token sequence, i.e., the length of the outputs of PDS is \( L \). \( L \) can be set by counting the lengths in the training set.

3.4. Decoder
After the PDS, we use the decoder to further capture token relationship. The outputs of the PDS can be seen as the representations corresponding to the tokens. So, we use self attention mechanism to capture the semantic relationship in the sequence. The decoder leverages a stack of attention blocks, and the keys, values and queries are the outputs of the previous block. After the decoder, we use a linear transformation to project the self attention based semantic representation, and softmax functions to compute probability distributions on the token vocabulary for each position.

3.5. Learning
For optimizing the parameters of the model, we minimize the position-wise cross entropy loss
\[
    \text{CE}(\theta) = -\frac{1}{NT} \sum_{(x,y) \in D} \sum_{i=1}^{L} \log P(y_{i}|x; \theta).
\]

### Table 1: The description of AISHELL-1. “Length” means the average number of tokens per utterance.

<table>
<thead>
<tr>
<th>Type</th>
<th>#Utter</th>
<th>#Hour</th>
<th>Length</th>
<th>#Speaker</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training</td>
<td>120,098</td>
<td>150</td>
<td>14.4</td>
<td>340</td>
</tr>
<tr>
<td>Development</td>
<td>14,326</td>
<td>18</td>
<td>14.3</td>
<td>40</td>
</tr>
<tr>
<td>Test</td>
<td>7,176</td>
<td>10</td>
<td>14.6</td>
<td>20</td>
</tr>
</tbody>
</table>

where \( D \) is the dataset which contains \( N \) pairs of speech and token sequence \( (x, y) \), \( L \) is the maximum length we pad to, and \( y_{i} \) is the token at position \( i \) in token sequence \( y \).

3.6. Inference
For decoding, we just select the token which has the highest probability at each position. Given an acoustic feature sequence, the predicted token at position \( i \) is
\[
    y_{i} = \arg \max_{y_{i}} P(y_{i}|x; \theta), \quad i = 1, \ldots, L,
\]
After prediction, the filler tokens “<eos>” at the tail of the sequence are removed.

4. Experiments
4.1. Datasets
We conduct experiments on a publicly available Chinese Mandarin corpus AISHELL-1 [11]. The dataset includes about 150 hours of speech for training, about 18 hours of speech for development, and about 10 hours speech for test. The speakers of training set, development set, and test set are not overlapped. All the recordings are in 16 kHz WAV format.

4.2. Experimental Setup
We use 80-dimension Mel-filter bank features (FBANK) as the input, which are extracted every 10ms with 25ms of frame length. The token vocabulary contains 4231 characters in training set and two special symbols, i.e., “<unk>” for unseen characters and “<eos>” as the filler of the tail of a token sequence.

The structure of the LASO model is shown in Fig. 1. Each layer of the two-layer subsampling CNN consists of 32 convolution filters with size 3 × 3, and the stride on time axis is 2. The activation functions of the CNN are ReLUs. All the attention blocks used in the model are the same. Both the encoder and the decoder have 6 attention blocks, i.e., \( N_{e} = N_{d} = 6 \) in Fig. 1. All the attention blocks have 8 heads. We compare different numbers of the attention blocks of PDS, i.e., \( N_{e} = 1, 2, 3 \) and 4, respectively. The intermediate dimensionality of the position-wise feedforward network is 2048, and the activation function is GLU. We train two types of LASO with different model dimensionalities \( D_{m} \). We refer to the model with \( D_{m} = 512 \) as LASO-base, and the model with \( D_{m} = 768 \) as LASO-big.

We re-implement Speech-Transformer as the baseline [5]. It uses the same CNN as our LASO architecture. Following their configuration, both encoder and the decoder have 6 layers. The dimensionality of the model is 512, and the intermediate dimensionality of the position-wise feedforward network is 2048. The number of heads of the multi-head attention is 8.

All models are trained with the same procedure. We use the Adam algorithm to train the model for 130 epochs. Each batch contains about 100 seconds of speech, and we accumulate gradients of 12 steps for simulating big batch [21]. We follow the warm-up learning rate schedule [4]:
\[
    \alpha = D_{m}^{-0.5} \cdot \min(step^{-0.5}, step \cdot \text{warmup}^{-1.5}),
\]
and the warm-up step is set to 12000. To avoiding overfitting, we set dropout rate to 0.1. We use SpecAugment [22] for data.
and retrain the two LASO models. We use factors 0.9 and 1.1 to perturb the speed of the audio and combine the augmented data with the original data. With speed perturbation, the CERs of LASO-base and LASO-big are further reduced to 6.8% and 6.4%, respectively.

We also show inference speed in Table 3. We can see that the latency of LASO models is much smaller than autoregressive models. The speed-up is about 50×. The non-autoregressive structure makes LASO do not need multi-pass forward computation in beam-search. And the feed-forward structure of LASO makes parallel computation efficient.

To better understand the behaviors of the PDS module, we visualize the attention scores of the 4-th attention block of the PDS in LASO-big. The attention scores are the average of the alignments of the meaningful tokens and the outputs of the encoder are from the upper-left to the bottom-right. For the filler token <eos>, the alignment is vague. Because no certain correspondence between the filler token and the outputs of the encoder exists. Because different head has different alignment in the multi-head attention, the averaged scores are not very sharp.

## 5. Conclusions and Future Works

In this paper, we propose a new non-autoregressive speech recognition model. We assume that speech signal contains the relationship among tokens implicitly, and token sequence can be generated without explicit language modeling. Based on this, we propose the LASO model. LASO forward propagates only one-pass for token generation, without beam-search. And because of the feedforward structure, parallel computation can be implemented efficiently, and time cost of inference can be significantly reduced. Experiments demonstrate that the proposed models have very low latency and promising performances. This work is the first result of the LASO model. In the future, we will investigate how to improve the performance of the LASO model by loss functions.

## 6. Acknowledgements

This work is supported by the National Key Research & Development Plan of China (No.2016YFB1001404), the National Natural Science Foundation of China (NSFC) (No.61831022, No.61901473, No.61771472, No.61773379) and Inria-CAS Joint Research Project (No.173211KYSB20170061 and No.173211KYSB20190049). This work is also (partially) funded by Huawei Noah's Ark Lab. We also thank the anonymous reviewers for their invaluable comments.
7. References


