An Improved Single Step Non-autoregressive Transformer for Automatic Speech Recognition

Ruchao Fan1, Wei Chu2, Peng Chang2, Jing Xiao2, Abeer Alwan1

1Dept. of Electrical and Computer Engineering, University of California, Los Angeles, USA
2PAII Inc., USA
{fanruchao,alwan}@g.ucla.edu, {chuwei29,changep805,xiaojing661}@pingan.com.cn

Abstract

Non-autoregressive mechanisms can significantly decrease inference time for speech transformers, especially when the single step variant is applied. Previous work on CTC alignment-based single step non-autoregressive transformer (CASS-NAT) has shown a large real time factor (RTF) improvement over autoregressive transformers (AT). In this work, we propose several methods to improve the accuracy of the end-to-end CASS-NAT, followed by performance analyses. First, convolution augmented self-attention blocks are applied to both the encoder and decoder modules. Second, we propose to expand the trigger mask (acoustic boundary) for each token to increase the robustness of CTC alignments. In addition, iterated loss functions are used to enhance the gradient update of low-layer parameters. Without using an external language model, the WERs of the improved CASS-NAT, when using the three methods, are 3.1%/7.2% on Librispeech test clean/other sets and the CER of error rate, and the performance is close to the autoregressive models.

Index Terms: non-autoregressive transformer, CTC alignment, token-level acoustic embedding, end-to-end ASR

1. Introduction

Transformers have been dominant in many sequence generation tasks, outperforming their recurrent neural network (RNN) counterparts in terms of both accuracy and speed for end-to-end systems [1–3]. However, the autoregressive (left-to-right) generation order slows down inference speed significantly. To accelerate the inference, non-autoregressive transformers (NAT) were proposed for the parallel generation of the output sequence. The idea is widely adopted in neural machine translation (NMT) [4–6], automatic speech recognition (ASR) [7–18], text-to-speech (TTS) [19, 20] and speech translation [21].

Current NAT models for ASR can be categorized into: (i) iterative NAT, and (ii) single step NAT, according to the number of iterations for sequence generation. Essentially, autoregressive models are also iterative-based since they use a left-to-right generation order and take N iterations to generate a sequence of length N. Hence, the idea of iterative NAT is to adopt a different generation order with less than N iterations to accelerate the inference. Chen et al. regarded the transformer decoder as a masked language model that first generates tokens with high confidence [7], while Higuchi et al. applied the same idea but based on the connectionist temporal classification (CTC) output [11, 16]. In addition, Fujita et al. used the idea of the insertion transformer from NMT to generate the output sequence with an arbitrary order [12]. Another recent effective method is using multiple decoders as refiners to do an iterative refinement based on CTC alignments [14]. Theoretically, the iterative NAT has a limited improvement of inference speed since multiple iterations are still needed to obtain a competitive result. In contrast, single step NAT, which attempts to generate the output sequence with only one iteration, can have a better speed up for inference. The idea is to substitute the word embedding in autoregressive models with an acoustic representation for each output token, assuming that language semantics can also be captured by acoustic representations [9, 10, 13].

Although various NAT methods were proposed for ASR, the WER performance still lags behind that of state-of-the-art autoregressive models. Therefore, based on our previous work [13], we propose several methods to improve the accuracy of CTC alignment-based single step NAT (CASS-NAT) with little inference speed loss. First, convolution augmented self-attention blocks are applied to both the encoder and decoder modules, while other work only considered using them in the encoder [22]. The second method is to expand the trigger mask (acoustic boundary) for each token to increase the robustness of the CTC alignment. Third, considering the wide use of iterated loss functions to train deep transformers [23–25], we apply iterated loss to enhance the gradient update of low-layer parameters for both the encoder and decoder modules. When no external language model is used, large improvements are observed on both the Librispeech [26] and Aishell1 [27] datasets in terms of error rate, and the performance is close to the autoregressive baseline. Additionally, we analyse the self-attention distributions and token-level acoustic embeddings in the decoders. We find a similar behaviour between the token-level acoustic embedding and word embedding, which explains why CASS-NAT performs similarly to autoregressive models.

The remainder of the paper is organized as follows. Section 2 briefly reviews the CASS-NAT and describes the proposed methods for improving the system. Section 3 describes the recognition experimental setup, followed by results and analyses in Section 4. Section 5 concludes the paper.

2. System Overview

2.1. Basic CASS-NAT Model

The CTC alignment-based single step non-autoregressive transformer (CASS-NAT) that we proposed in [13] is modified based on the CTC/Attention hybrid architecture [28] to be non-autoregressive. Fig.1 shows four major modules in the CASS-NAT: encoder, token acoustic extractor (TAE), self-attention de-
coder (SAD) and mixed-attention decoder (MAD).

Suppose the input sequence is $X = \{x_1, ..., x_T\}$ and ground truth is $Y = \{y_1, ..., y_T\}$, the encoder extracts high level acoustic representations $H$ from speech features $X$, followed by a CTC loss function. The role of CTC is to obtain an alignment over CTC output space to offer auxiliary information for the token acoustic extractor. Specifically, given an alignment $Z = \{z_1, ..., z_T\}$, we can estimate an acoustic segment for each token $u$ as $\{z_{u-1+1}, ..., z_u\}$, as well as the number of tokens in $Z$. The segment is equivalent to the trigger mask in Fig. 1, which is used for self-attention computation. The information obtained from the CTC alignment is then used to extract token-level acoustic embeddings that replace word embeddings existing in autoregressive transformers (AT). The self-attention and mixed-attention decoders are finally used to model the relationship between tokens, where the MAD has access to the acoustic representations $H$, while the SAD does not.

The framework is trained by jointly optimizing a CTC loss function on the encoder side ($L_{CTC}$) and a cross entropy (CE) loss function on the decoder side ($L_{dec}$) so that the final loss function ($L_{joint}$) is defined as:

$$L_{joint} = \lambda \log \sum_{Z \in q} \prod_{i=1}^{T} P(z_i | X) + \log \prod_{u=1}^{U} P(y_u | z_{u-1+1:t_u}, X)$$

where $P$ is the probability, $\lambda$ is the task ratio of the CTC loss, and $q$ is a set of alignments that can be mapped to the ground truth $Y$.

2.2. Convolution Augmented Self-attention Block

Self-attention is the most basic layer in speech transformers, and is formulated as:

$$Attn(Q, K, V, M) = \text{Softmax} \left( \frac{Q K^T}{\sqrt{d_k}} \right) \otimes M \cdot V$$

where $Q \in R^{n_q \times d_k}$, $K \in R^{n_k \times d_k}$, $V \in R^{n_v \times d_v}$, and $M \in R^{n_q \times n_k}$ are the queries, keys, values and mask matrix, respectively. From the equation, it can be seen that self-attention considers global information across the sequence, but ignores local details. To alleviate this problem, convolution augmented self-attention blocks are proposed to emphasise the modelling of local dependencies of the input sequence in the encoder [22,29]. Different from previous work, we apply the convolution augmented self-attention blocks in the mixed-attention decoder as well as the encoder. Specifically, the feed-forward layer is decomposed into two sub-layers to be placed at the beginning and the end of the block. A convolution layer similar to that in [22] is inserted after the self-attention layer except that we empirically use layer normalization instead of batch normalization. The final computation in the $i^{th}$ MAD can be formulated as:

$$s_i = s_i + \frac{1}{2} \text{FFN}(s_i)$$

$$s_i = s_i + \text{LN}(Attn(s_i, s_i, BiMask))$$

$$s_i = s_i + \text{Conv}(s_i)$$

$$s_i = s_i + \text{LN}(Attn(s'', H, H, BiMask))$$

$$o_i = \text{LN}(s'' + \frac{1}{2} \text{FFN}(s''))$$

where LN indicates layer normalization and FFN is the feed-forward layer. BiMask stands for a bidirectional mask.

In addition, different from the usage of relative positional encoding in [22,30], we consider a maximum length of relative position $k$ as [31]. Therefore, $2k + 1$ position embedding are learned to represent the relative position between $[-k, k]$.

2.3. Trigger Mask Expansion

The quality of token-level acoustic embedding relies on the accuracy of the trigger mask, which is mapped from the CTC alignment. Although the CTC loss function is used to optimize the alignment, there are still errors when doing forced alignment over CTC output space, leading to an inaccurate trigger mask. In order to compensate the inaccuracy of token-level acoustic embedding extraction, we expand the trigger mask to include contextual frames for each token. For example, suppose the contextual frame size is one, the acoustic boundary of token $U$ becomes $\{z_{u-1}, ..., z_{u+1}\}$. The trigger mask will then be expanded by one in the subsequent acoustic embedding extraction.

2.4. Iterated Loss

Deep transformers always suffer from gradient vanishing, especially for parameters that are distant from the output layers. Iterated loss are proposed to add additional loss functions after each layer to boost the gradient update [23,32]. Recent work proposed iterated CTC loss to improve the performance of CTC [25]. In [24], iterated CE loss is used for training deep transformer-based acoustic models. Since CTC and CE loss functions are jointly optimized in the CASS-NAT framework, we integrate iterated CTC and CE loss functions into Eq.1 so that the parameters in different layers can be updated at the same scale. We find this strategy to be more effective for CASS-NAT than AT models. Let $L_{dec} = \log \sum_{u=1}^{U} P(y_u | z_{u-1:t_u}, X)$ and $L_{CTC} = \log \sum_{z \in q} \prod_{i=1}^{T} P(z_i | X)$, the objective function is re-written as:

$$L_{joint} = \lambda_{CE} L_{dec} + (1 - \lambda_{CE}) L_{middle}$$

where $\lambda_{CE}$ and $\lambda_{CTC}$ are task ratios. Middle and final indicate the layer position of the inserted loss functions.
3. Experimental Setup

The experiments were conducted on three datasets: the 960-hour LibriSpeech English corpus [26], the 178-hour Aishell1 Mandarin corpus [27] and the 50-hour OGI Kids corpus (English) [33]. All experiments used 80-dim Mel-filter bank features, computed every 10ms with a 25ms Hamming window. Features of every 3 consecutive frames were concatenated to form a 240-dim feature vector as the input. The sets of output labels consist of 5k word pieces obtained by the SentencePiece method [34] for Librispeech and OGI. 4230 Chinese characters were obtained from the training set for the Aishell1 dataset.

A CTC/Attention AT baseline was first trained with the architecture \((N_e = 10, N_f = 5, d_{ff} = 2048, n_h = 8, d_{at} = 512)\) for Librispeech, \((N_e = 12, N_d = 6, d_{ff} = 2048, n_h = 4, d_{at} = 256)\) for the Aishell1 and \((N_e = 8, N_d = 4, d_{ff} = 2048, n_h = 4, d_{at} = 256)\) for the OGI. When training the CASS-NAT, the encoder was initialized with the AT encoder for faster convergence as in [35]. The decoder in the AT baseline was replaced by 1-block token-level acoustic extractor, 3-block self-attention decoder and 4-block mixed-attention decoder. The maximum length of relative position \(k\) was set to 20 and it is set to 10 for the Aishell1 and OGI. The evaluation of the real time factor (RTF) was conducted using an NVIDIA Tesla V100 GPU with batch size of one.

Each of the two convolution layers in Fig.1 has 64 filters with a kernel size of 3 and a stride of 2, leading to a 4x frame-rate reduction. The same learning schedule in [13] was adopted. Layer normalization, dropout with rate of 0.1 and label smoothing with a penalty of 0.1 were all applied as the common strategies for training a transformer. We also applied SpecAug [36] for fair comparisons with results in previous literature. We additionally applied speed-perturbation for the Aishell1 dataset for fair comparisons with results in previous literature. We additionally applied speed-perturbation for the Aishell1 dataset for fair comparisons with results in previous literature. The insertion projec-

### Table 1: WERs of the proposed methods for improving CASS-NAT on Librispeech. No external language model is used. SpecAug is used in all configurations. WERR is the incremental relative WER improvement on the test-other data.

<table>
<thead>
<tr>
<th>Model w/o LM</th>
<th>dev-clean</th>
<th>dev-other</th>
<th>test-clean</th>
<th>test-other</th>
<th>WERR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conformer AT</td>
<td>2.7</td>
<td>7.2</td>
<td>3.0</td>
<td>7.0</td>
<td></td>
</tr>
<tr>
<td>CASS-NAT</td>
<td>3.7</td>
<td>9.2</td>
<td>3.8</td>
<td>9.1</td>
<td></td>
</tr>
<tr>
<td>+ Conv-aug Enc.</td>
<td>3.1</td>
<td>7.9</td>
<td>3.3</td>
<td>7.9</td>
<td>13.2%</td>
</tr>
<tr>
<td>+ Conv-aug Dec.</td>
<td>3.0</td>
<td>7.8</td>
<td>3.1</td>
<td>7.6</td>
<td>3.8%</td>
</tr>
<tr>
<td>+ Tri. Mask Exp.</td>
<td>3.0</td>
<td>7.6</td>
<td>3.1</td>
<td>7.5</td>
<td>1.3%</td>
</tr>
<tr>
<td>+ Iterated CTC</td>
<td>2.8</td>
<td>7.3</td>
<td>3.1</td>
<td>7.3</td>
<td>2.7%</td>
</tr>
<tr>
<td>+ Iterated CE</td>
<td>2.8</td>
<td>7.3</td>
<td>3.1</td>
<td>7.2</td>
<td>1.4%</td>
</tr>
</tbody>
</table>

### Table 2: A comparison of error rates and RTFs with previously published results. RTFs of previous work are missing because the authors did not report them, or the machines used to test RTFs are different. †: use SpecAug. ⋆: use speed perturbation.

<table>
<thead>
<tr>
<th>Librispeech (WER)</th>
<th>Model</th>
<th>test clean</th>
<th>test other</th>
<th>RTF test</th>
<th>Previous work (NAT)</th>
<th>Model</th>
<th>test clean</th>
<th>test other</th>
<th>RTF test</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>ST-NAT [10]†</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>w/o 6.6</td>
<td>6.9</td>
<td>7.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>w/o 6.2</td>
<td>6.7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Insertion-NAT [12]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5.3</td>
<td>5.9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enhanced-NAT [15]†</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>w/o 5.3</td>
<td>5.2</td>
<td>5.8</td>
<td></td>
<td>0.011</td>
</tr>
<tr>
<td>BERT-LASO [18]⋆</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>w/o 5.3</td>
<td>5.3</td>
<td>5.8</td>
<td></td>
<td>0.011</td>
</tr>
<tr>
<td>CASS-NAT [13]†</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>w/o 5.3</td>
<td>5.4</td>
<td>5.8</td>
<td></td>
<td>0.011</td>
</tr>
<tr>
<td>Improved CASS-NAT†</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>w/ 4.8</td>
<td>5.2</td>
<td>5.4</td>
<td>0.200</td>
<td></td>
</tr>
</tbody>
</table>

### 4. Results and Analyses

#### 4.1. Results on Librispeech and Aishell1

Experiments are first conducted on the Librispeech corpus by incrementally adding the proposed methods based on the original CASS-NAT [13] as shown in Table 1. First, when convolution augmented self-attention blocks are applied to the encoder, the WER on the test-other set has a 13.2% relative improvement compared to the CASS-NAT baseline. When the convolution augmented blocks are also applied to the decoder, the WER drops further by a 3.8% relative improvement. An 1.3% relative WER improvement is observed for the trigger mask expansion method. When using the iterated loss function to both the encoder and decoder, we obtain an incremental 2.7% and 1.4% WER improvements, respectively. The final result has less than a 3% increase in relative WER compared to the AT baseline, which used a conformer structure. In the next sub-section, we will explain why the improved CASS-NAT can achieve a performance that is close to its autoregressive counterpart.

The proposed three methods are also used to train an improved CASS-NAT on the Aishell1 dataset. The final WER and real time factor (RTF) comparisons with previously published results are shown in Table 2, including both the Librispeech and Aishell1 data. Using the Librispeech dataset, the WER on test-other for the proposed methods has a 21% relative improvement over the original CASS-NAT with little WER degradation (from 0.010 to 0.014). This RTF still has a 36x speed up compared to the AT baseline when no external LM is used. When using an external LM during inference, CASS-NAT does not benefit from LM as much as the AT baseline does, which was also reported in [13]. Using the Aishell1 dataset, the RTF speed up is not as large as in Librispeech. The reason may be that the speed up of NAT benefits longer utterances; however, there are more short utterances in Aishell1. In addition, we use AT baseline for ranking alignments, which may increase the computational cost. Although we do not carefully tune the hyper-parameters on Aishell1, the character error rate (CER) on the test set still improves from 5.8% to 5.4%, which is close to the AT baseline.
4.2. Attention and Embedding Visualizations

To analyse the modelling of token-level acoustic embeddings, we choose the first utterance in the Librispeech train-clean-100 subset and plot the self-attention weights in the last block of the self-attention and mixed-attention decoders. The weights of the last 4 heads are shown in Fig.2. We can see from the figure that most of the heads learn a monotonic alignment between the token-level acoustic embeddings, indicating that each token relies more on adjacent tokens, which is similar to the idea of word embedding using continuous bag of words (CBOW) and skip-gram [37]. The monotonic alignment also shows the usefulness of the relative positional encoding because distant tokens with close semantic similarity have low attention weights.

To further investigate how the token-level acoustic embedding behaves, we extracted such embeddings from the first 3000 utterances in the train-clean-100 Librispeech subset. Each acoustic embedding has its own word piece ground truth. For each word piece, the embeddings are averaged to represent the final acoustic embedding. Using the same idea of visualizing word embedding, we randomly choose three word pieces and find their four closest embeddings using the cosine similarity distance. The 12 embeddings are reduced to a 2-dimensional space using principal component analyses (PCA) and are then plotted. Two examples are shown in Fig.3. The figure shows that the token-level acoustic embedding can learn not only the acoustic similarity, but also the word-piece level semantic similarity. For example, for the word piece _ROME, the closest tokens contain _FRANCE_ representing cities and countries, and _ROSE_ which has a similar pronunciation. The same situation can be seen in the lower part of the Fig.3 where each cluster has the same part-of-speech words, such as _MAY_ versus _SHOULD_ and _REMARKABLE_ versus _FORMIDABLE_. This behaviour of token-level acoustic embedding is very similar to word embedding, indicating that they can also be used to capture semantics. This may explain why the improved CASS-NAT has a similar performance to the AT baseline.

### 4.3. NAT for Child Speech

<table>
<thead>
<tr>
<th></th>
<th>dev</th>
<th>test</th>
<th>RTF on test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conformer AT</td>
<td>1.8</td>
<td>2.5</td>
<td>0.081</td>
</tr>
<tr>
<td>Improved CASS-NAT</td>
<td>2.2</td>
<td>2.6</td>
<td>0.019</td>
</tr>
</tbody>
</table>

In this section, we conduct experiments on the scripted part of the OGI kids corpus with the data partition in [38] except that 10% of the test set is chosen to be a development set for early stopping. The results are shown in Table 3. Since the OGI dataset has a fixed language pattern, the distributions of training and test sets are close. Although we trained the model within 15 epochs, the WER remains very competitive. Nevertheless, the conclusion of our improvements to CASS-NAT still holds and it has an impressive speed up than autoregressive models, which may be suitable for child ASR-based educational applications.

5. Summary and Conclusion

This paper presented three methods to improve the WER performance of CTC alignment-based single step NAT (CASS-NAT), followed by performance analyses. First, convolution augmented self-attention blocks were applied to the encoder and decoder modules. Second, the trigger mask was expanded for each token to compensate for the inaccuracy of CTC alignments. Third, iterated loss functions were used to enhance the gradient update of low-layer parameters. When using the three methods without external language models, we achieved a 7%–21% relative WER/CEIR improvement over the original CASS-NAT on the Librispeech and Aishell1 dataset with little RTF degradation. The WER performance was worse within 5%, in relative terms, than the autoregressive baseline, but maintained a 10–40x speed up. Moreover, attention weights and embedding visualization showed that the token-level acoustic embedding had similar behaviors with word embedding, explaining why the CASS-NAT performed similarly to AT.
6. References


