Unified Streaming and Non-streaming Two-pass End-to-end Model for Speech Recognition

统一的流式和非流式双通道端到端语音识别模型

*Binbin Zhang1, Di Wu1, Zhuoyuan Yao2, Xiong Wang2, Fan Yu2, Chao Yang1, Liyong Guo1, Yaguang Hu1, Lei Xie2, Xin Lei1*

*张彬彬1，吴迪1，姚卓元2，王雄2，范宇2，杨超1，郭立勇1，胡亚光1，谢磊2，辛磊1*

1Mobvoi Inc，Beijing, China

2Audio, Speech and Language Processing Group (ASLP@NPU), School of Computer Science, Northwestern Polytechnical University, Xi’an, China

binbinzhang@mobvoi.com

# 摘要

In this paper, we present a novel two-pass approach to unify streaming and non-streaming end-to-end (E2E) speech recognition in a single model. Our model adopts the hybrid CTC/attention architecture, in which the conformer layers in the encoder are modified. We propose a dynamic chunk-based attention strategy to allow arbitrary right context length. At inference time, the CTC decoder generates n-best hypotheses in a streaming way. The inference latency could be easily controlled by only changing the chunk size. The CTC hypotheses are then rescored by the attention decoder to get the final result. This efficient rescoring process causes very little sentence-level latency. Our experiments on the open 170-hour AISHELL-1 dataset show that, the proposed method can unify the streaming and non-streaming model simply and efficiently. On the AISHELL-1 test set, our unified model achieves 5.60% relative character error rate (CER) reduction in non-streaming ASR compared to a standard non-streaming transformer. The same model achieves 5.42% CER with 640ms latency in a streaming ASR system.

**Index Terms**: streaming speech recognition, two-pass, dynamic chunk, U2

在本文中，我们提出了一种新的两遍解码方法，将流式和非流式端到端（E2E）语音识别统一到一个模型中。该模型采用了混合CTC/attention结构，对编码器中的conformer层进行了修改。我们提出了一种基于动态块的注意策略来允许任意的右侧上下文长度。在推理时，CTC解码器以流方式产生n个最佳假设。只需改变块大小就可以很容易地控制推理延迟。然后由注意解码器对CTC假设进行重新排序，得到最终结果。这种高效的重新排序过程产生的句子级延迟非常小。在开放的170小时AISHELL-1数据集上的实验表明，该方法可以简单有效地统一流式模型和非流式模型。在AISHELL-1测试集上，与标准的非流式transformer相比，我们的统一模型在非流式ASR中实现了5.60%的相对字符错误率（CER）降低。在流式ASR系统中，该模型以640ms的延迟达到5.42%的CER。

**关键词**：流式语音识别，二次解码，动态块，U2

2020.12.10 arxiv

# 1.     介绍

近几年来，端到端（E2E）模型在语音识别中得到了越来越多的关注。E2E模型将声学、发音和语言模型结合到一个单一的神经网络中，并显示出与传统ASR系统相比具有竞争力的结果。主要有三种流行的端到端方法，即CTC[1,2]、递归神经网络变换器（RNN-T）[3,4]和基于attention的编解码器（AED）[5,6,7]。这两种模型在识别精度、应用场景等方面都有其优点和局限性，并且已经付出了很多努力来比较这些模型[8]，或者将其中的一些模型组合起来[9[[1]](#footnote-1)，10]。

虽然这些模型在非流式应用程序中有很好的性能，但要使模型以流方式工作通常需要大量的改进工作或导致大量的精度降低，为此已经做了大量的工作。对于RNN-T，提出了一种two pass[10[[2]](#footnote-2)，11[[3]](#footnote-3)]方法来缩小与非流式的*Listen, Attend, Spell*（LAS）模型的精度差距。对于AED，人们首先提出了硬单调注意[12[[4]](#footnote-4)]来单调对齐AED模型的输入和输出，然后它可以以流的方式工作，同样的思想，提出了单调分块注意（MoChA）[13[[5]](#footnote-5)]和单调多头注意[14[[6]](#footnote-6)]来进一步提高单调注意的性能和稳定性。

近年来，将非流式和流式语音识别模型统一为一个模型的研究也越来越受到重视。一些基于transducer的模型，如Y-model[15[[7]](#footnote-7)]和UNIVERSAL ASR[16[[8]](#footnote-8)]就是为此目的而设计的，并取得了良好的性能。统一模型不仅缩小了流模型与非流模型的精度差距，而且减轻了模型开发、训练和部署的负担。

在这项工作中，我们提出了一个新的框架U2来统一非流式和流式语音识别。我们的框架是基于混合CTC/attention体系结构与conformer。训练过程简单，避免了RNN-T模型的复杂技巧和不稳定性问题。为了支持流式传输，我们修改了conformer块，同时只带来很少的性能下降。此外，通过使用动态块训练策略，我们的框架允许用户在推理时控制延迟。我们的结果显示U2在Aishell-1上达到了最先进的流精度。

# 2.     相关工程

文献[9]中的混合CTC/attention端到端ASR利用训练中CTC和attention解码器的损失来实现快速收敛，提高AED模型的鲁棒性。然而，在解码过程中，它将注意力得分和CTC得分结合起来进行联合解码。这两个分数只有收到完整语音后才能计算，这决定了它是一种无法流式工作的模型。

论文[10]提出了two pass的RNN-T模型，其精度与LAS模型相当。然而，RNN-T训练非常消耗内存[17[[9]](#footnote-9)]，因此我们不能使用大batch进行训练，这导致训练速度非常慢，性能也很差。此外，RNN-T训练也不稳定，文[18[[10]](#footnote-10)]中提出了CTC预训练，文[19]中提出了交叉熵（CE）预训练来辅助RNN-T训练，这也是一个棘手而复杂的问题。这两个原因增加了RNN-T在语音识别中的应用难度，特别是对于缺乏计算和研究资源的小公司。文[10]提出了一种三步训练法来解决这一问题，使训练过程更加复杂。

对于统一的非流式和流式模型，Y-model在训练时使用可变上下文，在推理时可以使用多个可选上下文。然而，可选择的上下文是在训练阶段预先定义的，并且上下文是根据编码器层数、卷积运算的内核大小精心设计的。Dual-model在训练和推理中只有一种流配置，如果需要另一种具有不同推理延迟的流模型，则需要对该模型进行完全再训练。另外，Y-model和Dual-model都是基于RNN-T的模型，它们与RNN-T有着相同的缺点。

我们提出的U2是一个基于CTC和AED的联合模型，由CTC和AED的联合损失和动态组块注意联合训练，不仅统一了非流模型和流模型，给出了一个很好的结果，而且大大简化了训练过程，在流式应用程序中动态地控制延迟和准确度之间的平衡。

# 3.     U2

## 3.1.    模型体系结构

图1显示了建议的two-pass体系结构。它包括三个部分，一个共享编码器，一个CTC解码器和一个注意力解码器。共享编码器由多个Transformer[20]或Conformer[21]编码器层。CTC解码器由线性层和log softmax层组成，训练时CTC损失函数应用于softmax输出。Attention解码器由多个transformer decoder层组成。我们可以使共享编码器只看到有限的右侧上下文，然后CTC解码器可以在第一遍中以流模式运行。在第二遍中，共享编码器和CTC解码器的输出可以以不同的方式使用。下面详细说明训练和解码过程。

图1：双通道CTC和AED联合架构

## 3.2.    训练

### 3.2.1.    联合损失函数

训练损失是CTC损失和AED损失的结合，如1所示，其中是x声学特征，y是相应的注释，LCTC(x, y)，LAED(x, y)分别是CTC和AED损失，λ是平衡CTC和AED损失重要性的超参数。不同于基于RNN-T的两次解码[10]，其中使用了三步过程来稳定训练，我们可以通过从零开始的综合损失直接训练我们的模型，这大大简化了我们的训练过程。如[9]所示，综合损失也有助于模型更快地收敛，具有更好的性能。

Lcombined(x, y) = λLCTC (x, y) + (1−λ) LAED (x, y)（1）

### 3.2.2.    动态组块训练

提出了一种动态块训练技术，统一了非流式模型和流式模型，实现了时延控制。

如前所述，我们的U2只能在共享编码器是流式时实现流式。在标准的Transformer编码器层中使用了完全的自我关注，如图2（a）所示，每次输入*t*都依赖于整个输入，绿色表示存在依赖性，而白色表示没有依赖性。流式传输的最简单方法是使输入*t*只看到自己和之前的输入，即左注意，看不到右上下文，如图2（b）所示，但是与全上下文模型相比，有非常大的退化。另一种常见的技术是有限的输入*t*只看到有限的右上下文*t*+1，*t*+2，…，*t*+*W*，其中*W*是每个编码器层的右上下文，并且总上下文通过所有编码器层累积，例如，如果我们有*N*个编码器层，每个层都有*W*右上下文，总上下文是*N\*W*。右上下文通常提高性能然而，与单纯的左注意相比，我们应该仔细设计每一层的层数和每一个右上下文来控制整个模型的最终右上下文，而使用一致编码层（conformer-encoder-layer）时，事情会变得更加困难，其中使用了右上下文随时间的卷积。我们采用了chunk attention，如图2（c）所示，我们将输入按固定的组块大小分成几个组块，深绿色表示当前组块，对于每个组块我们都有输入[t+1，t+2，…，t+c]，每个组块都依赖于它自己和之前的所有组块。然后编码器的整个延迟取决于块大小，这很容易控制和实现。我们可以使用固定的块大小来训练模型，我们称之为静态块训练，并使用相同的块进行解码。

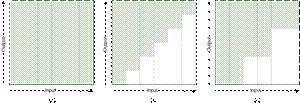
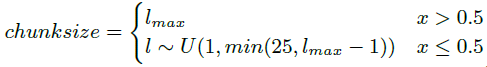


图2：完全注意，左注意，组块注意

基于统一E2E模型的思想，我们进一步提出了一种动态组块训练方法。在训练中，我们可以对不同的batch使用动态块大小，动态块大小的范围是从1到最大语句长度的均匀分布，即attention是从left context attention到full context attention的变化；模型在变化的chunk size上学习到不同信息；学习如何在提供有限右侧上下文时准确进行预测。对于流模型，我们将大小为1到25的块称为流块，对于无流模型，将大小为最大话语长度的块称为无流块。然而，这种方法的结果还不够好，因此接下来我们将在训练过程中改变块大小的分布如下。

 (2)

如式2所示，在训练过程中，每批x从0到1.0取样，lmax为当前批的最大话语长度，U为均匀分布。因此，块大小的分布发生了变化，一半是非流式使用完整块，另一半是1到25用于流式。

我们以后的实验将表明，这是一种简单而有效的方法，通过动态块大小训练的模型与静态块训练相比具有相当的性能。

除了这个批处理级别的方法，我们还尝试了epoch级别的方法——前半个epoch使用完整块，后半个epoch使用流块。但这些策略并不奏效。

### 3.2.3.    因果卷积

conformer中的卷积单元同时考虑了左右上下文。总的右侧上下文取决于卷积层的上下文和conformer层的堆栈数。因此，这种结构不仅带来了额外的延迟，而且破坏了基于组块的注意的好处，即延迟独立于网络结构，在推理时只能由组块控制。为了解决这个问题，我们使用了因果卷积[15]。

## 3.3.    解码

共享编码器逐块处理音频。较大的块大小通常意味着更高的延迟和更好的准确性，最大延迟与一个块的帧数成正比。正确的解码块大小取决于特定的任务要求。

CTC解码器以流方式输出一遍解码结果假设。在输入结束后，attention解码器使用全上下文注意来获得更好的结果。这里探讨了两种不同的模式：

•attention解码器模式。CTC结果在此模式下忽略。Attention解码器利用共享编码器输出计算attention，以自回归的方式产生输出。

•重新扫描模式。在教师强制模式下，attention解码器以共享编码器的输出为核心，对CTC的n个最佳假设进行验证。最终结果采用最佳重扫描假设。该模型避免了自回归过程，实现了较好的实时因子（RTF）。此外，还可以对CTC评分进行加权组合，以简单的方式得到更好的结果。

SCORESfinal = λ \* SCORESCTC + SCORESattention（3）

为了得到更好的结果，在重扫描模式解码过程中增加了ctc加权分数，如式3所示，我们后面的实验将表明这对解码结果总是有利的。

# 4.     实验

为了评估我们提出的U2，我们在开源汉语普通话语音语料库AISHELL-1[22]上进行了实验，该语料库包含150小时的训练集、10小时的开发集和5小时的测试集，测试集共包含7176条语句。所有的实验都使用了wenet[1]端到端语音识别工具箱。

我们使用先进的ASR网络作为共享编码器，解码器部分与传统的transformer解码器相同。Conformer在transformer的基础上增加了卷积模块，可以对局部和全局上下文捕获进行建模，在不同的ASR任务中都能得到更好的结果。对于conformer模型的动态组块训练，实验中用因果卷积代替了传统卷积，使en编码器独立于右侧上下文。

## 4.1.    aishell-1

对于AISHELL-1，我们使用80维log-mel滤波器组（FBank）拼接三维基音作为特征，该特征在25ms的窗口上计算，以10ms作为偏移。我们用0.9，1.0，1.1对整个数据进行速度扰动，产生3倍的速度变化。SpecAugment[23]应用了2个具有最大频率掩码（F=50）的频率掩码和2个具有最大时间掩码（T=50）的时间掩码。编码器前端采用两个卷积子采样层，核尺寸3\*3，步长2，共4次子采样。对于编码器，我们使用了12个conformer层和4个多头注意。对于注意解码器，我们使用了6个transformer层和4个多头注意。transformer和conformer都采用256注意维和2048前馈维。采用累积梯度法稳定训练，每四步更新一次参数。为了防止过拟合，在编码器层和解码器层分别采用了attention dropout、前馈dropout 和标签平滑正则化。我们使用Adam优化器和transformer学习率调度，以及25000个预热步骤来训练模型。此外，我们通过平均前10个最好的模型得到最终的模型，这些模型在训练时对dev集的损失较小。

表1：解码方法比较

|  |  |  |  |
| --- | --- | --- | --- |
| 解码方法 | CTC重量 | RTF | CER |
| 注意力解码器 | / | 0.197 | 5.00 |
| ctc前缀波束搜索 | / | / | 5.02 |
| Attention rescoring | 0.0 | / | 4.77 |
| Attention rescoring | 0.5 | 0.082 | 4.72 |

### 4.1.1.    解码方法

首先，我们在非流模型上探讨了不同的解码方法，在训练中使用全上下文和标准卷积核大小为15的conformer，以确保CTC和AED解码器都能给出合理的结果。对于AED解码器，我们使用波束10进行解码。我们使用前缀波束搜索CTC，用于生成前n个不同的假设，以便以后重打分。

如表1所示，attention rescoring结果优于CTC前缀波束搜索和注意解码器结果，这超出了我们的预期。

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 表2：动态组块训练与静态组块训练   |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | 培训方法 | 解码模式 |  | 解码块大小 | | |  | | full | 16 | 8 | 4 | 1 | | 静态语块训练，静态语块推理 | 注意力解码器 | 5.35 | 5.95 | 5.99 | 6.15 | 6.36 | | ctc前缀波束搜索 | 5.18 | 6.30 | 6.50 | 6.69 | 6.73 | | Attention rescoring | 4.86 | 5.55 | 5.78 | 6.06 | 6.02 | | 动态语块训练，静态语块推理 | 注意力解码器 | 5.82 | 5.99 | 6.13 | 6.29 | 6.60 | | ctc前缀波束搜索 | 5.51 | 6.23 | 6.57 | 6.92 | 7.83 | | Attention rescoring | 5.05 | 5.48 | 5.66 | 5.85 | 6.40 | |

通过对CTC前缀波束搜索和attention rescoring解码结果的分析，发现CTC前缀波束搜索产生的很多错误结果都可以通过attention rescoring得到纠正，而在一些好的情况下，rescoring后引起了错误的纠正，这意味着CTC在某些情况下起着重要的作用。因此，我们在重打分/rescoring过程中增加了CTC权重，如等式3所示。我们测试了0.1到0.9之间不同的CTC权重，它们都有助于我们的实验中的注意力重打分，0.5是最稳定的权重。这里我们只展示了当CTC权重为0.5时的结果，正如我们看到的，当与CTC权重相结合时，CER可以进一步降低到4.72。据我们所知，这是AISHELL-1上公布的最好的结果。在以后的实验中，0.5是rescoring模式的默认CTC权重。

由于标准的注意力解码器是以一种自回归的方式运行的，这是很费时的，而注意力重打分仅仅使用注意力解码器进行重打分，因此它可以并行处理，理论上应该更快。因此，本文还研究了注意解码器的RTF和注意重打分方法，并在Pytorch中采用了单线程的方式进行解码。如表1所示，在解码过程中，通过注意重打分，我们比注意解码器的速度提高了2.40倍。

综上所述，我们可以看到注意力重打分的速度更快，更准确。

### 4.1.2.    动态块评价

如前所述，在动态组块训练中使用因果卷积来统一非流式和流式模型，并且使用了8的核大小，这是之前实验的一半，因为这里的模型只能看到左上下文。

为了与静态块训练进行比较，我们用full/16/8/4/1训练了五种不同的静态块大小模型，然后以相同的块大小作为基线进行解码。我们只训练了一个统一的模型，上面提到的动态块策略如方程2所示。结果如表2所示，由于这是我们系统的最终性能，所以我们主要关注这里的注意力重打分结果。从表中可以看出，动态块训练模型在全块和块1上有一定的退化，这两个点分别是无限延迟和无延迟动态块的两个边界点。我们猜想在统一模型中学习边界信息更困难。然而，当组块大小为16/8/4时，我们发现与静态组块训练模型相比，动态组块策略稍有提高，这意味着在这种情况下，动态组块策略通过改变组块训练有利于统一模型。

总体而言，动态组块训练模型是可比较的静态组块训练模型，因此我们的U2框架可以通过two-pass解码和动态组块训练将非流模型和流模型统一为一个模型。

### 4.1.3.    与其他解决方案的比较

表3列出了几个已发布的在AISHELL-1测试集上的流式解决方案，包括Sync-Transformer [24]、SCAMA[25]和MMA[14]。∆是在我们的U2解码结束时，注意力重打分引入的额外延迟，但它足够快，正如我们之前所说的，它可以并行到一批计算中，如[10，15]中所分析的，它是50-100ms。我们可以看到，我们的U2已经远远超过了其他解决方案，只增加了很小的延迟。

表3：与其他流式解决方案的比较

|  |  |  |  |
| --- | --- | --- | --- |
| 模型 | 参数（M） | 延迟（毫秒） | CER |
| ync-Transformer [24] | / | 400 | 8.91 |
| SCAMA[25] | 43 | 600 | 7.39 |
| MMA[14] | / | 640 | 6.60 |
| 我们的U2 | 47 | 640+∆ | 5.42 |

# 5.     结论

我们提出了一个框架来训练一个单一的模型，它可以在流式和全上下文两种方式下进行识别。该框架无需复杂的训练过程，可以直接、稳定地进行训练。采用一种快速的加权重评分方法，在不增加延迟的情况下获得完整的上下文性能。我们还提出了一种基于动态分块的策略来提高模型的性能，并在推理时方便地权衡延迟和准确性。

# 6.     工具书类

[1] A. Graves, S. Fern´andez, F. Gomez, and J. Schmidhuber, “Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks,” in Proceedings of the 23rd international conference on Machine learning, 2006, pp. 369–376.

[2] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen et al., “Deep speech 2: End-to-end speech recognition in english and mandarin,” in International conference on machine learning, 2016, pp. 173–182.

[3] A. Graves, “Sequence transduction with recurrent neural networks,” arXiv preprint arXiv:1211.3711, 2012.

[4] A. Graves, A.-r. Mohamed, and G. Hinton, “Speech recognition with deep recurrent neural networks,” in 2013 IEEE international conference on acoustics, speech and signal processing. IEEE, 2013, pp. 6645–6649.

[5] J. Chorowski, D. Bahdanau, K. Cho, and Y. Bengio, “End-to-end continuous speech recognition using attention-based recurrent nn: First results,” arXiv preprint arXiv:1412.1602, 2014.

[6] W. Chan, N. Jaitly, Q. V. Le, and O. Vinyals, “Listen, attend and spell,” arXiv preprint arXiv:1508.01211, 2015.

[7] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, “Attention-based models for speech recognition,” in Advances in neural information processing systems, 2015, pp. 577– 585.

[8] R. Prabhavalkar, K. Rao, T. N. Sainath, B. Li, L. Johnson, and N. Jaitly, “A comparison of sequence-to-sequence models for speech recognition.” in Interspeech, 2017, pp. 939–943.

[9] S. Kim, T. Hori, and S. Watanabe, “Joint ctc-attention based end-to-end speech recognition using multi-task learning,” in 2017 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE, 2017, pp. 4835–4839.

[10] T. N. Sainath, R. Pang, D. Rybach, Y. He, R. Prabhavalkar, W. Li, M. Visontai, Q. Liang, T. Strohman, Y. Wu et al., “Two-pass endto- end speech recognition,” arXiv preprint arXiv:1908.10992, 2019.

[11] T. N. Sainath, Y. He, B. Li, A. Narayanan, R. Pang, A. Bruguier, S.-y. Chang, W. Li, R. Alvarez, Z. Chen et al., “A streaming on-device end-to-end model surpassing server-side conventional model quality and latency,” in ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2020, pp. 6059–6063.

[12] C. Raffel, M.-T. Luong, P. J. Liu, R. J.Weiss, and D. Eck, “Online and linear-time attention by enforcing monotonic alignments,” arXiv preprint arXiv:1704.00784, 2017.

[13] C.-C. Chiu and C. Raffel, “Monotonic chunkwise attention,” arXiv preprint arXiv:1712.05382, 2017.

[14] H. Inaguma, M. Mimura, and T. Kawahara, “Enhancing monotonic multihead attention for streaming asr,” arXiv preprint arXiv:2005.09394, 2020.

[15] A. Tripathi, J. Kim, Q. Zhang, H. Lu, and H. Sak, “Transformer transducer: One model unifying streaming and non-streaming speech recognition,” arXiv preprint arXiv:2010.03192, 2020.

[16] J. Yu, W. Han, A. Gulati, C.-C. Chiu, B. Li, T. N. Sainath, Y. Wu, and R. Pang, “Universal asr: Unify and improve streaming asr with full-context modeling,” arXiv preprint arXiv:2010.06030, 2020.

[17] J. Li, R. Zhao, H. Hu, and Y. Gong, “Improving rnn transducer modeling for end-to-end speech recognition,” in 2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU). IEEE, 2019, pp. 114–121.

[18] K. Rao, H. Sak, and R. Prabhavalkar, “Exploring architectures, data and units for streaming end-to-end speech recognition with rnn-transducer,” in 2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU). IEEE, 2017, pp. 193–199.

[19] H. Hu, R. Zhao, J. Li, L. Lu, and Y. Gong, “Exploring pre-training with alignments for rnn transducer based end-to-end speech recognition,” in ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2020, pp. 7079–7083.

[20] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” in Advances in neural information processing systems, 2017, pp. 5998–6008.

[21] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, J. Yu,W. Han, S. Wang, Z. Zhang, Y. Wu et al., “Conformer: Convolutionaugmented transformer for speech recognition,” arXiv preprint arXiv:2005.08100, 2020.

[22] H. Bu, J. Du, X. Na, B. Wu, and H. Zheng, “Aishell-1: An open-source mandarin speech corpus and a speech recognition baseline,” in 2017 20th Conference of the Oriental Chapter of the International Coordinating Committee on Speech Databases and Speech I/O Systems and Assessment (O-COCOSDA). IEEE, 2017, pp. 1–5.

[23] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, “Specaugment: A simple data augmentation method for automatic speech recognition,” arXiv preprint arXiv:1904.08779, 2019.

[24] Z. Tian, J. Yi, Y. Bai, J. Tao, S. Zhang, and Z.Wen, “Synchronous transformers for end-to-end speech recognition,” in ICASSP 2020- 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2020, pp. 7884–7888.

[25] S. Zhang, Z. Gao, H. Luo, M. Lei, J. Gao, Z. Yan, and L. Xie, “Streaming chunk-aware multihead attention for online end-toend speech recognition,” arXiv preprint arXiv:2006.01712, 2020.

1. *Joint ctc-attention based end-to-end speech recognition using multi-task learning. 2017.* [↑](#footnote-ref-1)
2. *Two-pass end-to-end speech recognition. 2019.* [↑](#footnote-ref-2)
3. *A streaming on-device end-to-end model surpassing server-side conventional model quality and latency. 2020.* [↑](#footnote-ref-3)
4. *Online and linear-time attention by enforcing monotonic alignments. 2017* [↑](#footnote-ref-4)
5. *Monotonic chunkwise attention. 2017.* [↑](#footnote-ref-5)
6. *Enhancing monotonic multihead attention for streaming asr. 2020.* [↑](#footnote-ref-6)
7. *Transformer transducer: One model unifying streaming and non-streaming speech recognition. 2020* [↑](#footnote-ref-7)
8. *Universal asr: Unify and improve streaming asr with full-context modeling. 2020* [↑](#footnote-ref-8)
9. *Improving rnn transducer modeling for end-to-end speech recognition. 2019.* [↑](#footnote-ref-9)
10. *Exploring architectures, data and units for streaming end-to-end speech recognition with rnn-transducer. 2017.* [↑](#footnote-ref-10)