IKFST：用于加速和提升精度的端到端自动语音识别IOO与KOO算法

Zhuoran Zhuang[*]♣ **Ye Chen**[*]♣ **Chao Luo**[*]♣ **Tian-Hao Zhang**[*]♠、**Xuewei Zhang**♣ **Jian Ma**♣ **Jiatong Shi**♡、**Wei Zhang**[†]♣ 飞猪阿里巴巴♠ 北京科技大学♡卡内基梅隆大学
zhichen.zw@alibaba-inc.com,
tianhaozhang@xs.ustb.edu.cn

arXiv:2601.00160v1 [cs.SD] 1 Jan 2026

摘要

端到端自动语音识别已成为学术界和工业界的主流范式。为提升识别性能，加权有限状态换能器（WFST）被广泛用于通过静态图组合整合声学模型和语言模型，从而实现稳健的解码和有效的错误纠正。然而，WFST解码依赖于针对CTC后验概率逐帧进行的自回归搜索，这严重限制了推理效率。为了建立WFST解码与CTC建模之间更为合理的兼容机制，我们系统地研究了CTC输出的两个基本组成部分，即空白帧和非空白帧，并发现一个关键规律：空白帧主要编码位置信息，而非空白帧承载语义内容。基于这一观察，我们提出仅保留一个（Keep-Only-One）和Insert-Only-One两种解码算法，明确利用空白帧和非空白帧的结构作用，在不牺牲识别准确率的前提下，显著加速基于WFST的推理。在大规模内部数据集、AISHELL-1和LibriSpeech数据集上的实验表明，该方法在大幅降低解码延迟的同时，达到了最先进的识别准确率，使现代语音识别系统能够实现真正高效且高性能的WFST解码。

1 引言

深度神经网络的最新进展显著提升了自动语音识别（ASR）的性能（Sainath 等人，2015；Dong 等人，2018；Wu 等人，2023b；Zhang 等，2024a；Zhou 等人，2024）。传统 ASR 系统依赖由专家设计组件构成的多阶段流程（JuangandRabiner，1991），而端到端（E2E）模型则通过直接映射来简化这一过程语音到文本（Prabhavalkar 等人，2024）。端到端 ASR 方法主要包括 CTC（Graves 等人，2006；Amodei 等人，2016；Yao 等，2025）、RNN-T（Graves，2012；Graves 和 Jaitly，2014）和基于注意力的编码器–解码器模型（Chan 等人，2016；Zhang 等，2023a,b），以及近期提出的混合形式，如 CTC/RNN-T 和 CTC/AED（Watanabe 等人，2017；Kim 等人，2017；Zhang 等，2024b）。在这些系统中，推理时通常会集成外部语言模型，以增强语言一致性并实现领域适应，常用方法包括 N-best 重排序或非自回归重打分（Chorowski 和 Jaitly，2017；Kannan 等人，2018；Sainath 等人，2021；Yao 等，2021）。

在各种解码框架中，加权有限状态换能器（WFST）凭借扎实的理论基础、灵活的拓扑组合能力和稳健的解码能力，长期以来一直是大型 ASR 系统的基础组件（Mohri 等人，2000；Hori 和 Nakamura，2022；Lv 等人，2021）。在这一框架中，CTC 模型提供帧级后验概率，随后通过维特比束搜索，使用经过修改的 WFST 对其进行解码（Miao 等人，2015；Laptev 等人，2022；Povey 等人，2016）。尽管这种方法有效且应用广泛，直接对完整后验序列执行 WFST 解码仍然需要耗费极高的计算资源，成为高效、可扩展部署的一大挑战。为缓解上述问题，Chen 等人（Chen 等人，2016）提出了基于格的音素同步解码（LSD）算法。Speech-LLaMA（Wu 等人，2023a）提出了一种平均策略，以保留这些潜在信息。PolyVoice（Dong 等人，2023）则系统地丢弃所有空白帧。基于类似的见解，尖峰窗口解码（SWD）算法进一步改进了这一理念：选择性地纳入高概率非空白尖峰周

[*]贡献相同。
[†]通讯作者。

围有限范围内的空白帧（Zhang 等，2025）。

综合以往研究的见解，我们得出一个统一的观察：即使剪枝掉大量空白帧，模型在 WFST 解码中仍能取得有竞争力、甚至更优的识别结果。这一观察促使我们深入研究空白帧和非空白帧内在的位置与语义特征，旨在找出能够实现最优解码的最精简且足够的帧表示。基于这一思路，我们提出了两种互补算法：Insert-Only-One（IOO）和 Keep-Only-One（KOO）。IOO算法首先丢弃所有由模型学习得到的概率性空白帧，然后在相邻的非空白帧之间策略性地插入一个确定性、由用户定义的空白帧，从而保留关键的过渡线索，同时消除冗余的时间信息。与此同时，KOO算法针对非空白区域内的冗余，通过在每个激活簇中有选择地保留一个具有代表性的尖峰，在不损害语义保真度的情况下有效压缩后验序列。值得注意的是，IOO算法适用范围广，可提升 CTC-FST 和 AED-FST 解码的性能。

我们在多种数据集上对所提出的 KOO 和 IOO 算法进行了广泛评估，以全面考察其有效性和泛化能力。实验采用了广泛使用的 AISHELL-1 普通话数据集（Bu 等人，2017）、英语 Librispeech 数据集（Panayotov 等人，2015），以及一个包含 65,000 小时数据的大规模内部数据集。我们首先构建了一个 CTC/AED 混合声学模型，并将其与 GPU 加速的 WFST 解码框架（Daniel 和 Kaldewey，2023）相结合，形成稳健的基础，实现了最先进水平（SOTA）的识别准确率。在此基础上，所提出的方法在推理速度和识别性能方面均取得了显著提升。这些结果验证了该方法在不同语言环境和数据规模下实现高效、高准确率解码的能力。

2 相关工作

2.1 基于 CTC 的 ASR 模型

CTC 是一种广泛采用的端到端 ASR 目标函数，因为它能够在无需进行帧级对齐的情况下训练模型。给定一个声学特征序列 X，我们将 Y 定义为对应的标签序列，其长度为 L。编码器会生成一系列隐藏表示：

$$H_{encoder} = \text{Encoder}(X). \tag{1}$$

CTC 通过允许空白符号和重复词元，引入中间对齐序列，从而实现声学帧与输出标签之间灵活的多对一映射。令$\mathcal{B}(\cdot)$表示移除空白符号和重复符号的映射；CTC 目标函数为：

$$\mathcal{L}_{CTC} = -\log \sum_{Z \in \mathcal{B}^{-1}(Y)} p(Z \mid H_{encoder}), \tag{2}$$

其中，对齐路径 Z 的概率是在时间步之间满足条件独立假设的情况下建模的。

2.2 混合 CTC/AED 算法

CTC 常与基于注意力的编码器–解码器（AED）结合使用。AED 解码器以编码器表示和先前生成的词元为条件：

$$H_{decoder} = \text{Decoder}(H_{encoder}, Y), \tag{3}$$

并使用交叉熵进行训练：

$$\mathcal{L}_{AED} = \text{CrossEntropy}(H_{decoder}, Y). \tag{4}$$

混合目标函数对这两种损失进行插值：

$$\mathcal{L} = \alpha \mathcal{L}_{CTC} + (1 - \alpha) \mathcal{L}_{AED}, \tag{5}$$

其中，α 是用于调整编码器与解码器之间权重比例的超参数，其取值范围从 $[0, 1]$，开始；本研究其余部分通常将其设为 0.1。

2.3 基于 WFST 的解码算法

我们在编码器端集成了一个基于加权有限状态转换器的解码模块，并按照标准 TLG 组合框架构建静态解码图。整个图通过词元、词典和语法转换器的顺序组合得到：

$$TLG = T \circ \min(\det(L \circ G)), \tag{6}$$

其中，L 和 G 分别表示词典有限状态转换器和语法有限状态转换器。运算符det、

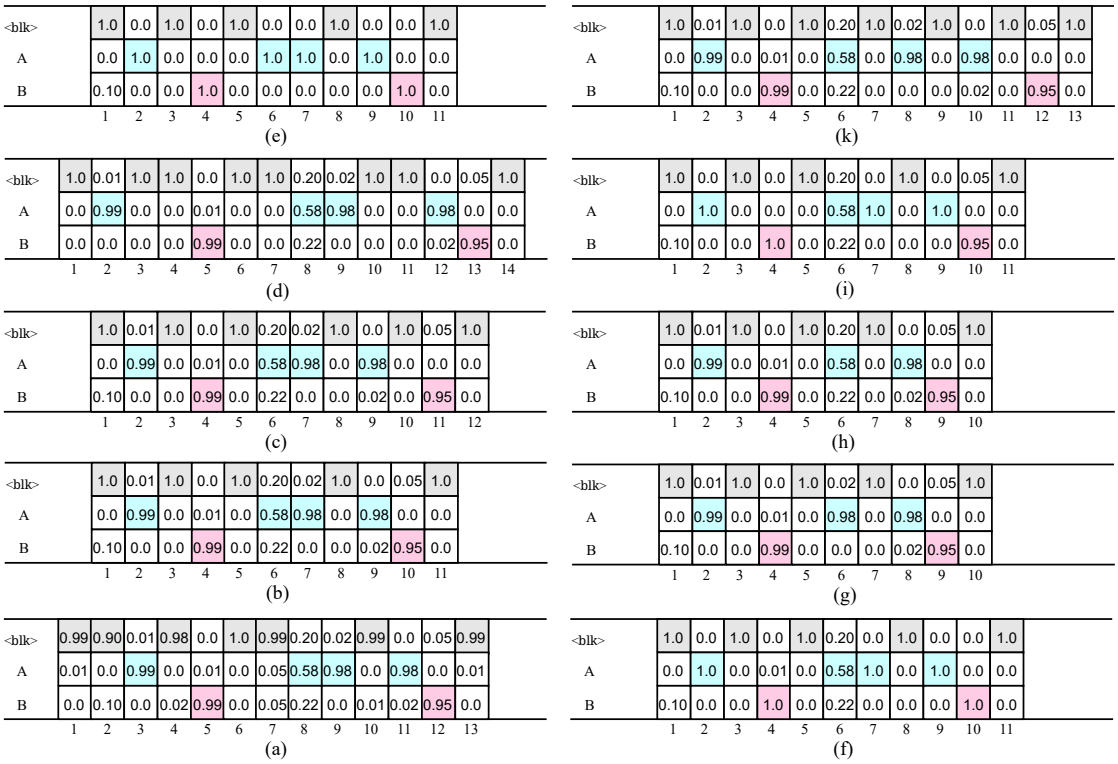

图1：(a) 表示使用密集帧进行的搜索；(b)-(d) 和 (e)-(f) 分别展示了在空白帧和非空白帧上使用 IOO 算法的情况；(g) 保留后验概率最高的帧，而 (h) 保留后验概率最低的帧；(i) 表示基于概率应用 KOO 算法；(k) 表示将 AED-FST 与 IOO 算法集成。

min和 ∘ 分别对应确定化、最小化和WFST组合。值得注意的是，我们采用了（T）中提出的GPU加速WFST解码器和紧凑型词元转换器Daniel和Kaldewey，2023），该方法在保持具有竞争力的解码准确率的同时，大幅降低了图构建的状态空间复杂度——从二阶指数级降至线性级。

3 方法论

本研究探讨将 CTC后验概率与 FST 集成，以在推理速度和识别准确率之间实现最佳权衡。我们系统地分析了 CTC后验中空白输出与非空白输出的不同特性，并据此提出了两种新算法：Insert-Only-One（IOO）和 Keep-Only-One（KOO）。结合这两种算法后，我们展现了处理速度和识别性能的显著提升。值得注意的是，使用 IOO算法是缓解 AED-FST 准确率严重下降的有效方法。

3.1 IOO 和 KOO 算法

在 CTC框架中，编码器会生成一系列 logit $H_{encoder}$，表示各时间步上的原始、未归一化词元分数。为获得对应的后验 logit P，需对词汇维度应用 softmax函数：

$$P = \text{SoftMax}(H_{encoder}). \tag{7}$$

这会在每一帧上为所有可能的词元（包括空白符号）生成一个概率分布。如图1 (a)所示，CTC模型生成的后验分布呈现出明显的尖峰状模式，其特点是大量高概率的空白预测与词元激活交错出现（图中以两个示例词元类别 A 和 B 表示）。如果以密集方式进行解码，即将完整的 CTC 后验序列直接提供给 FST，解码效率会大幅下降。这种下降是由数量庞大的空白占主导的帧导致的，它们极大地扩展了

有效搜索空间，并在 FST 遍历过程中带来显著的计算开销。

因此，我们引入 IOO算法来缓解这一问题。如图1(b) 所示，IOO算法会以受控方式插入定制空白帧。这一策略既保留了空白符号所带来的理想时间分离，又缓解了原始 CTC后验序列中空白区域密集出现所导致的解码效率低下问题。具体而言，给定 CTC后验序列 P，IOO 算法会移除所有原始空白帧，并将包含一个或多个连续空白预测的位置替换为单个定制空白分布。每个定制空白帧都定义为一个独热概率向量，其中空白标记（索引为 0）的后验概率设为 1.0，其余所有词元的概率均设为 0.0。形式上，插入的空白帧表示为：

$$p_{blk} = [1.0,\, 0.0,\, 0.0,\, \ldots,\, 0.0] \in \mathbb{R}^{|\mathcal{V}|}, \quad (8)$$

其中，$|\mathcal{V}|$ 表示词汇表大小。这种替换不仅消除了空白片段密集出现所带来的效率低下问题，还引入了一种一致且确定的空白表示，从而提高了FST解码的稳定性。最终经 IOO增强的后验序列记为 P_{IOO}。

作为对所提出的 IOO算法（用于优化冗余空白帧）的补充，我们进一步引入 KOO算法，以精炼 CTC后验序列中的非空白部分。给定帧级概率序列P_{IOO}，KOO算法沿原始时间轴直接识别非空白区域。将帧 t 处预测的词元记为$c_t = \arg\max_{v\in\mathcal{V}} P_{t,\,v}$。随后，通过将具有相同非空白预测结果的连续时间步分组，构造非空白块 B：

$$B_k = \left\{ t_k^{\text{start}}, t_k^{\text{start}} + 1, \ldots, t_k^{\text{end}} \right\}, \quad (9)$$

$$\text{s.t. } c_t = c_k \neq \text{blank},\ \forall t \in B_k, \quad (10)$$

其中，k 按时间顺序枚举非空白块，t_k^{start}对应属于第 k 个块的第一个时间步。重要的是，KOO 必须作用于完整序列（c_1，c_2，…，c_T），不能移除空白帧；过早丢弃空白帧可能会人为地合并两个不相邻的帧，导致时间邻接关系错误。

如图1（g）和图1（h）所示，对于每个块 B_k，KOO根据最大概率或最小概率策略选择一个代表帧：

$$f^* = \begin{cases} \arg\max\limits_{t\in B_k} P_{t,\,c_k}, & \text{max strategy}, \\ \arg\min\limits_{t\in B_k} P_{t,\,c_k}, & \text{min strategy}. \end{cases} \quad (11)$$

收集与所选索引对应的 logit，即可得到精炼后的非空白序列：

$$L_{\text{nb}} = \left[P_{f_1^*}, P_{f_2^*}, \ldots, P_{f_k^*} \right] \in \mathbb{R}^{K\times|\mathcal{V}|}. \quad (12)$$

所得序列 L_{nb} 保留了原始 CTC输出的时间结构，同时消除了 FST解码过程中由重复非空白帧引入的累积误差。通过为每个非空白区域仅保留一个代表帧，该序列能够实现更准确、更可靠的解码结果。

当将 KOO 应用于经过 IOO 处理的概率序列 P_{IOO} 时，最终后验 P_{final}融合了 IOO 的位置信息和 KOO 的语义信息，从而保留K 非空白帧。由于 IOO算法在每个保留的非空白帧之后最多插入一个定制的空白帧，因此最终序列的长度满足：

$$|P_{\text{final}}| \leq 2K + 1. \quad (13)$$

由于 K 显著小于 CTC 后验序列中的原始时间步数 T，参与后续 FST解码的帧数大幅减少。因此，FST搜索步数总量也按比例减少，从而显著提升解码效率。针对 CTC-FST，所提出的 IOO 和 KOO 过程的伪代码见算法1 （附录 A）。

3.2 AED-IOO集成

尽管基于AED的语音识别模型近来已成为主流范式，并在许多基准测试中取得了最先进性能，但我们的实验揭示了一个重要局限：将原始 AED后验序列直接输入基于FST的解码器，会导致结果不尽如人意。我们认为，尽

管AED模型中没有明确的空白概念，FST搜索过程仍依赖于相邻帧之间某种形式的时间分离，才能有效运行。

受这一观察的启发，我们通过在每个解码器生成的帧后插入一个定制空白帧，将 IOO 流程扩展到 AED。最终的输出序列如图1(k)所示。具体而言，这相当于在 AED 后验序列中加入确定性的空白分布，其形式与 CTC 侧使用的定制空白相同。实验表明，这一改动始终能够提高识别准确率，说明引入空白符号引起的时间分离能够为 FST 解码带来有益的正则化效果，即使是在原生不使用空白符号的 AED 框架中也是如此。所提出的 AED-FST IOO算法的伪代码见算法2（附录B）。

3.3 TLG 图优化

传统的基于 CTC 构建 TLG 解码图的方法总结于公式6。然而，将这一构建过程应用于大型语言模型时，往往会生成规模极大的 TLG 图，进而导致内存消耗大幅增加、存储需求上升，以及推理过程的成本更高。为应对这些挑战，本文在图构建流程中加入了一个权重推送步骤，位于$\det$ 和$\min$ 操作之间。通过将路径权重向较早的状态移动，该操作能够在不牺牲解码准确率的前提下，有效地提前剪除低概率路径。因此，有效搜索空间和运行效率均得到显著改善。最终的静态图构建过程形式化如下：

$$TLG = T \circ \min(\text{pushing}(\det(L \circ G))) \quad (14)$$

4 实验

4.1 数据集

我们在 LibriSpeech、AISHELL-1 和大规模内部数据集上开展实验。LibriSpeech 是一个包含 960 小时英语语料库的数据集，而 AISHELL-1 是一个包含 178 小时普通话语料库的数据集，采样率为 16 kHz。我们的内部数据集同样是普通话语音语料库，采样率为 8 kHz。它包含约 65,000 小时的带标签训练数据，而测试集由从呼入和呼出电话中随机采样的约 6,000 条短话语组成。

4.2 实验设置

4.2.1 声学模型设置

本文构建的声学模型采用多任务混合CTC/AED结构，其中编码器和解码器模型分别基于Zipformer和Transformer模型。在所有数据集上，编码器均采用（Yao 等，2023）中的大型Zipformer；解码器则基于标准Transformer模型，包含6个Transformer模块。声学模型最终规模为0.22B个参数，编码器相关的关键配置汇总于表3（附录C）中。

4.2.2 语言模型设置

对于 AISHELL-1 和 LibriSpeech 数据集，语言模型仅使用各自训练集中的文本构建。具体而言，使用 SRILM工具包[1]训练 5-gram语言模型。在内部数据集上，我们使用约 4300 万条普通话对话文本数据构建语言模型。第3节介绍的运算组合、det、min 和权重推送均使用 OpenFst工具 [2]实现。

4.2.3 训练、推理与评估

训练阶段提取 80 维滤波器组作为语音特征，帧长为 25 ms，帧移为 10 ms。为增强数据，采用语音速度扰动（Ko 等人，2015），扰动系数为 0.9、1.0 和 1.1。此外，还采用 SpecAugment（Park 等人，2019）策略来提升模型的鲁棒性。所有模型均在 16 张 NVIDIA Tesla H200 GPU 上使用混合精度训练。推理时，所有计算均统一在配备 16 核 CPU 和 32GB RAM 的 Tesla T4 GPU（16GB）上执行。在推理阶段，使用标准字符错误率（CER）评估识别性能，该指标通过计算预测序列与对应真实转录文本之间的 Levenshtein 距离（Levenshtein，1966）得出。为进一步加快解码，所有实验均采用批大小 5。

[1]https://www.sri.com/platform/srilm [2] https://www.openfst.org/twiki/bin/view/FST/WebHome

表1：内部数据集上的实验结果。箭头表示数值越高或越低越理想。

ID	模型	解码类型	字符错误率 (%) ↓	加速比 ↑
A1	CTC Zipformer	CTC 贪心搜索	3.27	3.45 ×
A2	AED Zipformer	AED 贪心搜索	3.21	0.87 ×
B1	A1 + 5-gram	稠密 CTC FST	3.09	1.00 ×
B2	A2 + 5-gram	稠密 AED FST	3.56	0.51 ×
C	B1 + LSD（Chen 等人，2016）	0.99空白阈值	3.12	1.18 ×
D	B1 + Speech-LLaMA（Wu 等人，2023a）	平均	3.10	1.29 ×
E	B1 +PolyVoice（Dong 等人，2023）	丢弃	4.71	1.99 ×
F	B1 + SWD（Zhang 等，2025）	$\{0, \pm 1\}$	3.08	1.47 ×
G1	B1 +IOO-B	$\{0, 1\}$	3.06	1.65 ×
G2	B1 +IOO-B	$\{0, 1, 2\}$	3.06	1.54 ×
G3	G1 + IOO-NB	$\{*\}$	3.78	1.72 ×
G4	G1 + IOO-NB	最大值$\{*\}$	3.75	1.75×
G5	G1 + KOO	$\min\{*\}$	3.06	1.67 ×
G6	G1 + KOO	最大值$\{*\}$	**3.05**	1.69 ×
H	B2 + IOO-B	$\{0, 1\}$	3.13	0.43 ×

4.3 实验结果

4.3.1 内部数据集实验。

表 1 对所提出的 IOO 和 KOO算法在内部数据集上的表现进行了定量评估，并与标准解码基线和具有代表性的启发式加速方法进行了比较。A1 行和 A2 行分别报告了 Zipformer 模型使用 CTC 和 AED输出进行贪心解码的结果，界定了简单贪心搜索下的性能范围。B1 行和 B2 行展示了使用 5-元语法稠密加权有限状态转换器解码得到的相应结果，作为主要的准确率-延迟参考。具体而言，B1 行（CTC Zipformer + 5-gram）确立了稠密基线，其字符错误率为 3.09%，归一化解码速度为 1.00×，所有后续加速结果均以此为衡量基准。

实验 C 重点评估将基于标签同步的算法与 TLG 解码图结合时的影响，该解码图采用 0.99 的空白阈值。LSD 算法基于以下假设进行应用：随着帧丢弃阈值的提高，识别准确率将接近原始稠密系统，但推理速度会相应降低。实验 D 和 E 分别对相邻非空白帧之间的空白帧取平均，以及丢弃所有空白帧。尽管这些方法确实提高了推理速度，但识别准确率也因此付出了更大的代价。

以 B1 基线为起点，G1 表明 IOO-B 可将解码速度提高 1.65×，且不会损失识别准确率。在 G2 中插入额外的空白帧并无改善，反而略微降低了速度；而 G3 中采用激进简化策略，将所有非空白帧替换为确定性的独热分布，则会导致识别准确率大幅下降。G4 采用了更保守的策略，在相同词元簇中保留后验概率最高的帧，从而减轻准确率损失，同时仍能提升速度。

实验G5和G6验证了KOO算法：每个词元仅保留一个非空白帧，便能显著加速，同时不降低准确率；其中，保留后验概率最高的帧效果最佳。这些结果表明，对于基于CTC的WFST解码，保留适当的空白帧并仅保留信息量最大的非空白帧，既能维持识别准确率，又能最大限度地提高计算效率。

实验H进一步证实了IOO在AED输出上的普适性，并有效缓解了直接进行WFST解码时观察到的准确率下降。总体而言，这些发现印证了一个关键见解：空白帧编码位置信息，而非空白帧承载语义内容。

我们进一步探讨了解码超参数对FST性能的影响。图中的分析2 表明，Beam值必须在剪枝与搜索广度之间取得平衡，才能维持准确率和速度；而格束可以设为固定值，不会影响性能。最大活动令牌数需要仔细调整，以避免搜索崩溃或内存使用过多，约5000个词元可提供

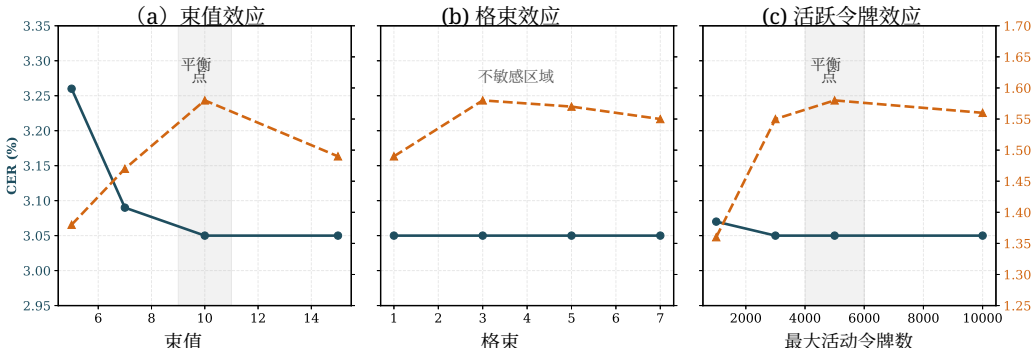

图 2：解码参数的敏感性分析。双轴图展示了 (a) 束值、(b) 格束和 (c) 最大活动令牌数对字符错误率（CER，实线青绿色）和解码加速比（虚线橙色）的影响。阴影区域表示研究中确定的最佳工作点，结果表明格束基本不敏感，而束值和活动令牌数则需要进行平衡调优。

一个有效的运行点。重要的是，这些结果证实，IOO–KOO框架的改进并不依赖于特定的参数选择：每个空白区域只需一个定制空白帧，并且每个尖峰只需保留概率最高的非空白帧，即可保持准确率，同时大幅加快解码速度。

4.3.2 开源数据集实验。

表2展示了 AISHELL-1 和 LibriSpeech 数据集上的实验结果。在这两个数据集上，我们的 IOO–KOO 框架均能持续提高解码效率，同时保持识别准确率。在 AISHELL-1 上，将 IOO 或 KOO 应用于 5-元语法 CTC系统（B1）后，字符错误率进一步降低：从 3.73%/3.94% 降至 3.71%/3.93%（G1）和 3.70%/3.92%（G6），同时解码速度提升超过 2.3 倍。在 LibriSpeech 数据集上也观察到类似趋势。与密集 CTC 基线 B1（1.95%/3.94%）相比，IOO（G1）和 KOO（G6）均保持了相当的准确率，且速度提升超过 2 倍。总体而言，普通话和英语语料库上的结果证实，IOO–KOO 对语言类型和语料规模具有稳健性。该框架始终能够显著加快解码，同时实现持平甚至更高的识别准确率，彰显了其在实际基于FST的自动语音识别部署中的应用价值。

5 讨论与分析

用于位置信息的 IOO 我们推测，在 FST 解码过程中，CTC 式方法中的空白帧发挥着双重功能：除吸收非语义声学变化外，还会——在时间轴上隐式编码位置信息，从而约束非空白令牌的相对顺序和间隔。这一假设得到了多项实证观察的支持。首先，在基于 CTC 的模型中，诸如“A–B–<blk>–<blk>–<blk>”、“<blk>–<blk>–<blk>–A–B”和“<blk>–A–<blk>–B–<blk>”这样的令牌序列，在语义上都等同于目标序列“AB”，体现了 CTC 不受对齐方式影响的特性。然而，我们的实验结果表明，大幅丢弃空白帧对性能的影响有限，而将其完全移除则会导致识别准确率严重下降。此外，尽管基于 AED 的模型在传统 FST 解码下通常会遭受显著的准确率损失，但采用所提出的 IOO算法后，其性能得到显著提升，因为该算法在解码过程中保留了必要的空白帧结构。这些发现表明，空白帧在 FST 解码中充当一种隐式的位置编码，使非空白令牌在时间上保持锚定，并令解码过程更加稳定。因此，在任意两个相邻的非空白帧之间至少插入一个空白帧，对于实现与 Dense-FST 解码相当的性能至关重要。这一认识从原理上解释了空白帧的必要性：它们不仅仅是非语义声学内容的占位符，还承载着不可或缺的位置信息，决定时间对齐并增强解码鲁棒性。

用于语义信息的 KOO KOO 算法通过选择性梳理语义上冗余的非空白帧来改进 FST 解码

表2：AISHELL-1 数据集上的实验结果。开源数据集的结果使用字符错误率进行评估，而 LibriSpeech 数据集使用词错误率进行评估。

模型	AISHELL-1			LibriSpeech		
	开发集（%）↓	测试集（%）↓	加速比 ↑	测试集-清晰（%）↓	测试集-其他（%）↓	加速比 ↑
CR-CTC（Yao 等，2025）	**3.69**	3.98	-	**1.88**	3.95	-
CIF-Transducer (Zhang 等，2024b)	4.1	4.3	-	-	-	-
Zipformer (Yao 等，2023)	4.03	4.28	-	1.96	4.08	-
Branchformer(Peng 等，2022)	4.19	4.43	-	2.4	5.3	-
E-Branchformer (Kim 等，2023)	4.2	4.5	-	2.4	4.6	-
Paraformer (Gao 等，2022)	4.7	5.1	-	-	-	-
Conformer(Gulati 等人，2020)	4.5	4.9	-	2.1	4.3	-
A1	3.98	4.19	3.63 ×	1.99	4.08	3.20 ×
A2	3.97	4.15	0.90 ×	2.94	4.15	0.89 ×
B1	3.73	3.94	1.00 ×	1.95	3.94	1.00 ×
B2	6.33	6.76	0.68 ×	3.77	6.98	0.55 ×
G1	3.71	3.93	2.37 ×	1.94	3.95	2.07 ×
G6	3.70	**3.92**	2.36 ×	1.94	**3.94**	2.06 ×
H	3.77	4.03	0.55 ×	2.01	4.18	0.49 ×

，从而在保留关键信息的同时抑制无关的声学变化。这种有针对性的剪枝为后续解码提供了更高质量的输入，并在效率和识别准确率方面均带来稳定提升。KOO 的有效性得到了三个关键观察结果的支持。首先，表1 （G5 和 G6）中最初观察到的性能提升在表 2 （G6）中得到了稳定复现，证明了所提方法的稳健性。其次，随着剪枝阈值从 0.8 提高到 0.99（附录 D），识别准确率单调提升，在 G10 中达到 3.06% 的相对增益，同时仍保持超过 1。6× 倍的推理速度提升。最后，KOO 在理论上与 CTC 的多对一映射特性高度一致。在训练过程中，相邻的相同非空白帧会在 CTC格中自然合并；而在解码过程中，诸如“A–A–<blk>–<blk>”、“<blk>–<blk>–A–A”以及“<blk>–A–A–<blk>”在语义上都等价于目标标签“A”。这种内在等价性为剪枝冗余非空白帧提供了合理的理论依据，同时不会降低解码正确性。

KOO 的隐藏优势 据我们所知，现有采用 TLG 图进行解码的端到端自动语音识别技术在构建静态 Token.fst 图时存在一个潜在问题（Miao 等人，2015；Laptev 等人，2022）：由于在非初始状态执行自环操作，连续非空白帧的概率会变得过高。为解决这一问题为抑制连续相同非空白帧导致的路径概率膨胀，KOO算法在 FST解码过程中仅选取后验概率最高的单帧。表1和2（G1 与 G6）进一步验证了 KOO 在解决路径概率膨胀问题上的有效性。这些发现提出了一种新颖而有效的范式，可在基于 FST 的解码过程中利用 CTC/AED 后验概率，并为进一步优化提供了新思路。

6 结论

本文深入探讨了 CTC/AED 输出的脉冲行为，并提出这样一个猜想：空白帧提供位置信息，而非空白帧携带对模型有益的语义信息。在此基础上，我们提出了两种互补算法，以提升基于 CTC/AED 的端到端自动语音识别系统的推理速度和识别准确率，分别称为 IOO算法和 KOO算法。通过重构空白帧与非空白帧序列，我们的方法能够更高效地与加权有限状态变换器集成，大幅减少解码帧数。此外，采用 IOO算法可以缓解 AED-FST 中准确率严重下降的问题。进一步地，我们在确定化和最小化步骤之间引入了权重推送优化，通过提前剪枝提高 TLG 搜索效率。在内部数据集、AISHELL-1 和 Librispeech 数据集上的实验结果表明，IOO算法和 KOO算法能够显著提升推理速度，同时甚至还能提高识别准确率。

局限性

尽管本研究表明解码效率和准确率均得到显著提升，但目前的评估主要集中于 AISHELL-1 和 LibriSpeech 等标准基准测试。因此，所提算法在更广泛的低资源语言和复杂声学环境中的鲁棒性仍有待进一步探索。

参考文献

Dario Amodei、Sundaram Ananthanarayanan、Rishita Anubhai、Jing Bai、Eric Battenberg 等另外 1 位作者。2016。Deep Speech 2：端到端英语和普通话语音识别。发表于第*33*届国际机器学习会议（*ICML*）论文集，第 173–182 页。

Hui Bu、Jiayu Du、Xingyu Na、Bengu Wu 和 Hao Zheng。2017。AISHELL-1：一个开源普通话语音语料库及语音识别基线。发表于*IEEE* 东方分会国际语音数据库与语音输入/输出系统及评估协调委员会会议论文集（*O-COCOSDA*），第 1–5 页。

William Chan、Navdeep Jaitly、Quoc Le 和 Oriol Vinyals。2016。Listen, Attend and Spell：一种用于大词汇量会话语音识别的神经网络。发表于 *IEEE* 国际声学、语音与信号处理会议（*ICASSP*），第 4960–4964 页。

Zhehuai Chen、Yan Zhuang、Yanmin Qian 和 Kai Yu。2016。基于CTC格的音素同步语音识别。*IEEE/ACM* 音频、语音与语言处理汇刊，25(1)：90–101。

Jan Chorowski 和 Navdeep Jaitly。2017。探索序列到序列模型中更优的解码与语言模型集成。载于*ISCA* 国际语音通信协会年会（*Interspeech*），第 523–527 页。

Galvez Daniel 和 Tim Kaldewey。2023。用于基于CTC的语音识别的 GPU 加速 WFST Beam 搜索解码器。载于*IEEE IEEE* 自动语音识别与理解研讨会（*ASRU*）论文集，第 1–7 页。

Linhao Dong、Shuang Xu 和 Bo Xu。2018。Speech-transformer：一种用于语音识别的无循环序列到序列模型。载于*IEEE IEEE* 声学、语音与信号处理国际会议（*ICASSP*），第 5884–5888 页。

Qian Dong、Ziyue Huang、Qiao Tian、Chen Xu、Tom Ko 及另外 1 位作者。2023 年。PolyVoice：用于语音到语音翻译的语言模型。*arXiv* 预印本 *arXiv:2306.02982*。

Zhifu Gao、Shiliang Zhang、Ian McLoughlin 和 Zhi-jie Yan。2022 年。Paraformer：用于非自回归端到端语音识别的快速且准确的并行 Transformer。载于国际语音通信协会年度会议（*Interspeech*），第 2063–2067 页。Alex Graves。2012 年。使用循环神经网络进行序列转导。*arXiv* 预印本 *arXiv:1211.3711*。Alex Graves、Santiago Fernandez、Faustino Gomez 和 Jurgen Schmidhuber。2006 年。连接主义时序分类：使用循环神经网络标注未分段序列数据。载于第 *23* 届国际机器学习会议（*ICML*）论文集，第 369–376 页。

Alex Graves 和 Navdeep Jaitly。2014。通过循环神经网络实现端到端语音识别。载于第*31*届国际机器学习会议（*ICML*）论文集，第1764–1772页。

Anmol Gulati、James Qin、Chung-Cheng Chiu、Niki Parmar、Yu Zhang 等。2020。Conformer：用于语音识别的卷积增强型Transformer。载于国际语音通信协会年度会议（*Interspeech*），第5036–5040页。

Takaaki Hori 和 Atsushi Nakamura。2022。使用加权有限状态转换器的语音识别算法。SpringerNature。

Biing-Hwang Juang 和 Lawrence R Rabiner。1991。用于语音识别的隐马尔可夫模型。计量学，33：251–272。

Anjuli Kannan、Yonghui Wu、Patrick Nguyen 等。2018。将外部语言模型融入序列到序列模型的分析。载于*IEEE*国际声学、语音与信号处理会议（*ICASSP*）。

Kwangyoun Kim、Felix Wu、Yifan Peng、Jing Pan、Prashant Sridhar 等另外 1 位作者。2023。E-branchformer：采用增强型合并的 Branchformer 语音识别模型。载于*IEEE* 口语语言技术研讨会（*SLT*）论文集，第 84–91 页。

Suyoun Kim、Takaaki Hori 和 Shinji Watanabe。2017。基于多任务学习的 CTC-注意力端到端语音识别。载于*IEEE* 国际声学、语音与信号处理会议（*ICASSP*），第 4835–4839 页。

Tom Ko、Vijayaditya Peddinti、Daniel Povey 和 San- jeev Khudanpur。2015 年。《用于语音识别的音频增强》。载于国际语音通信协会年度会议（*Interspeech*），第 3586–3589 页。

Aleksandr Laptev、Somshubra Majumdar 和 Boris Ginsburg。2022 年。《通过新的 WFST 拓扑实现 CTC 变体》。载于国际语音通信协会年度会议（*Inter- speech*），第 1041–1045 页。

Vladimir I Levenshtein。1966。能够纠正删除、插入和反转的二进制码。《苏联物理学文献》，10(8)：707–710。

Hang Lv、Zhehuai Chen、Hainan Xu、Daniel Povey、Lei Xie 和 Sanjeev Khudanpur。2021。用于自动语音识别的异步基于 WFST 的解码器。在 *IEEE* 国际声学、语音与信号处理会议（*ICASSP*）上，第 6019–6023 页。

Yajie Miao、Mohammad Gowayyed 和 Florian Metze。2015。Eesen：使用深度循环神经网络模型和基于 WFST 的解码进行端到端语音识别。在*IEEE* 自动语音识别与理解研讨会论文集（*ASRU*）上，第 167–174 页。

Mehryar Mohri、Fernando CN Pereira 和 Michael D Riley。2000。加权有限状态转换器库的设计原则。理论计算机科学，231(1)：17–32。

Vassil Panayotov、Guoguo Chen、Daniel Povey 和 San- jeev Khudanpur。2015。《LibriSpeech：基于公有领域有声读物的自动语音识别语料库》。载于*IEEE* 国际声学、语音与信号处理会议（*ICASSP*），第 5206–5210 页。

Daniel S Park、William Chan、Yu Zhang、Chung-Cheng Chiu、Barret Zoph 等人。2019。《SpecAugment：一种用于自动语音识别的简单数据增强方法》。载于国际语音通信协会年度会议（*Interspeech*），第 2613–2617 页。

Yifan Peng、Siddharth Dalmia、Ian Lane 和 Shinji Watanabe。2022。《Branchformer：用于捕获语音识别与理解中局部和全局上下文的并行 MLP 注意力架构》。载于 机器学习国际会议论文集（*ICML*），第 17627–17643 页。

Daniel Povey、Vijayaditya Peddinti、Daniel Galvez、Pegah Ghahremani、Vimal Manohar 等 1 人。2016。基于无格 MMI 的纯序列训练神经网络，用于 ASR。载于国际语音通信协会 *(Interspeech)* 年会，第 2751–2755 页。

Rohit Prabhavalkar、Takaaki Hori、TaraN Sainath、Ralf Schluter 和 Shinji Watanabe。2024。端到端语音识别：综述。*IEEE/ACM* 音频、语音与语言处理汇刊，32:325–351。

Tara N Sainath、Yanzhang He、ArunNarayanan、Rami Botros、Ruoming Pang 等其他 1 位作者。2021。一种高效的流式、非循环、端侧端到端模型，并改进了罕见词建模。载于国际语音通信协会年度会议（*Interspeech*），第 1777–1781 页。Tara N Sainath、Brian Kingsbury、George Saon、Hagen Soltau、Abdel-rahman Mohamed 等其他 1 位作者。2015。用于大规模语音任务的深度卷积神经网络。神经网络，64:39–48。Shinji Watanabe、Takaaki Hori、Suyoun Kim、John R Hershey 和 Tomoki Hayashi。2017。用于端到端语音识别的混合式 CTC/注意力架构。*IEEE* 信号处理精选主题期刊，11(8):1240–1253。

Jian Wu、Yashesh Gaur、Zhuo Chen、Long Zhou 和 Yujia Zhu。2023a。关于用于语音转文本与大语言模型集成的仅解码器架构。载于*IEEE* 自动语音识别与理解研讨会（*ASRU*）论文集，第 1–8 页。

Jian Wu、Yashesh Gaur、Zhuo Chen、Long Zhou、Yujia Zhu 等 1 位作者。2023b。关于用于语音转文本和大语言模型集成的仅解码器架构。载于 *IEEE* 自动语音识别与理解研讨会论文集（*ASRU*），第 1–8 页。

Zhao Yao、Long Guo、Xing Yang、Wei Kang、Fangjun Kuang 等 1 位作者。2023。Zipformer：一种更快、更出色的自动语音识别编码器。载于第十二届国际学习表征会议论文集（*ICLR*）。

Zhao Yao、Wei Kang、Xing Yang、Fangjun Kuang、Long Guo 等 1 位作者。2025。Cr-ctc：在 CTC 上进行一致性正则化，以提升语音识别效果。载于第十四届国际学习表征会议论文集（*ICLR*）。

Zhuo-You Yao、Di Wu、Xiong Wang、Bin-Bin Zhang、Fan Yu 等 1 位作者。2021。Wenet：面向生产环境的流式与非流式端到端语音识别工具包。载于国际语音通信协会（*ISCA*）年度会议（*Interspeech*），第 4054–4058 页。

Tian-Hao Zhang、Qi Liu、Xinyuan Qian、Song-Lu Chen、Feng Chen 和 Xu-Cheng Yin。2023a。用于自动语音识别的自卷积。载于*IEEE* 国际声学、语音与信号处理会议（*ICASSP*），第 1–5 页。

Tian-Hao Zhang、Xinyuan Qian、Feng Chen 和 Xu-Cheng Yin。2024a。用于自动语音识别的传输与聚合自注意力。载于国际语音通信协会年度会议（*Interspeech*），第 227–231 页。

Tian-Hao Zhang、HanQin、Zhi-Hao Lai、Song-LuChen、Qi Liu 和 Xu-Cheng Yin。2023b。通过

声学与语义协同解码。在*ISCA* 国际语音通信协会年会（*Interspeech*），第 914–918 页。

Tian-Hao Zhang、Dong Zhou、Guo Zhong、Jiatong Zhou 和 Bo Li。2024b。CIF-T：一种新颖的基于 CIF 的自动语音识别转导器架构。载于*IEEE* 国际声学、语音与信号处理会议（*ICASSP*），第 10531–10535 页。

Wei Zhang、Tian-Hao Zhang、Cheng Luo、Hang Zhou、Chao Yang 等。2025。突破尖峰：用于加速且精准的自动语音识别的尖峰窗口解码。载于*IEEE* 国际声学、语音与信号处理会议（*ICASSP*）。

Jiatong Zhou、Sheng Zhao、Yang Liu、Wei Zeng、Yujia Chen 等。2024。KNN-CTC：通过检索 CTC 伪标签增强 ASR。载于*IEEE* 国际声学、语音与信号处理会议（*ICASSP*），第 11006–11010 页。

一种用于 CTC-FST 的 IOO

算法1 用于 CTC 的 IOO 和 KOO算法

要求： 后验概率 $P \in \mathbb{R}^{T \times V}$
确保： 处理后的序列 H_{final}

函数 PROCESSPOSTERIORSCTC(P)
  $B \leftarrow$队列 ▷初始化帧块队列
  $i, j, k \leftarrow 0$
  $koo \leftarrow$false ▷初始化koo设置
  while $i < T$ **do**
    $p_c \leftarrow -1$ ▷ 将上一个 c 初始化为*-1*
    $c, p \leftarrow$argmax($P[i]$), max($P[i]$)
    若 $c \neq$ p_c **则**
      $p_c \leftarrow c$
      $b \leftarrow [(P[i], p)]$ ▷ 初始化子块 b
      $j \leftarrow i + 1$
      当 $j < T \wedge$ argmax($P[j]$) $= c$ **执行**
        b.追加((($P[j]$, max($P[j]$)))
        $j \leftarrow j + 1$
      B.推入(c, b) ▷推入 b队列 B
      $i \leftarrow j$ ▷ 跳转到下一个块
    else
      $i \leftarrow i + 1$
  $H_{final} \leftarrow []$
  $v_{blk} \leftarrow [1.0, 0, \ldots, 0]$ ▷ 独热空白向量
  H_{final}.追加(v_{blk}) ▷ 第一个自定义帧
  当 B 不为空 **时**
    $b \leftarrow B$.弹出
    如果 $k = 0 \wedge b$.键 $= 0$ **则**
      $k \leftarrow k + 1$
      继续
    如果 b.键 $=$0**则**
      阶段 1：Insert-Only-One（IOO）
      H_{final}.追加(v_{blk}) ▷ 使用 *IOO*
    否则如果 非koo **则**
      H_{final}.追加(b.值) ▷ 不使用 *KOO*
    else
      阶段 2：Keep-Only-One（KOO）
      $f^* \leftarrow$选择最大概率帧(b.值)
      H_{final}.追加(f^*)
  返回 H_{final}

B IOO 的 AED-FST

算法2 用于 AED 的 IOO 算法

要求： 后验概率 $P \in \mathbb{R}^{T \times V}$
确保： 处理后的序列 H_{final}

函数PROCESSPOSTERIORSAED(P)
  $i \leftarrow 0$
  $H_{final} \leftarrow []$
  $v_{blk} \leftarrow [1.0, 0, \ldots, 0]$ ▷独热空白向量
  H_{final}.追加(v_{blk})
  当 $i < T$**时执行**
    H_{final}.追加($P[i]$)
    H_{final}.追加(v_{blk})
    $i \leftarrow i + 1$
  返回 H_{final}

C 编码器配置

表3：实验中使用的 Zipformer 配置。

参数	值
层数	2, 2, 4, 6, 4, 2
前馈网络维度	512、768、1536、2048、1536、768
编码器维度	192, 256, 512, 768, 512, 256
编码器未屏蔽维度	192、192、256、320、256、192

D KOO 非空白帧

表4：内部数据集上非空白帧阈值的结果。

ID	非空白阈值	字符错误率(%) ↓	加速比 ↑
G3	$\{*\}$	3.78	1.72 ×
G7	$\{* \geq 0.8\}$	3.77	1.73 ×
G8	$\{* \geq 0.90\}$	3.29	1.62 ×
G9	$\{* \geq 0.95\}$	3.13	1.64 ×
G10	$\{* \geq 0.99\}$	3.06	1.61 ×