短期记忆卷积

Grzegorz Stefa´nski、Krzysztof Arendt、Paweł Daniluk、Bartłomiej Jasik、Artur Szu maczuk三星波兰研发院{g.stefanski、k.arendt、p.daniluk、b.jasik、a.szumaczuk}@samsung.com

arXiv:2302.04331v1 [cs.LG] 8 Feb 2023

摘要

对于许多实际应用而言，时间序列信号的实时处理至关重要。实时处理的理念在音频领域尤为重要，因为人类声音感知对感知信号中的任何干扰都十分敏感，尤其是听觉和视觉模态之间的延迟。深度学习（DL）模型的兴起使信号处理的局面更加复杂。尽管这些模型的输出质量通常优于标准数字信号处理方法，但更高的延迟削弱了这一优势。本文提出了一种名为短期记忆卷积（STMC）及其转置对应方法的新方法，用于降低推理延迟和内存消耗。STMC 的主要优势是延迟较低，可与长短期记忆网络（LSTM）相媲美。此外，基于 STMC 的模型训练速度更快、稳定性更高，因为该方法完全基于卷积神经网络（CNN）。本研究展示了该方法在用于语音分离任务的 U-Net 模型以及用于声学场景分类（ASC）任务的 GhostNet 模型中的应用。在语音分离任务中，我们将推理时间缩短了 5 倍，延迟降低了 2 倍，同时未影响输出质量。对于 ASC 任务，在保持原有准确率的同时，推理时间最高可缩短至原来的四分之一。

1 引言

卷积神经网络（CNN）已成为深度学习（DL）中的主流模型类型之一。最突出的应用领域是计算机视觉，例如图像分类（Rawat & Wang，2017年）、目标检测（Zhao 等，2019）或图像分割（Minaee 等，2021年）。

卷积神经网络模型在某些信号处理任务中证明了其有效性，尤其适用于无需长期上下文的任务，例如语音增强（Sun 等，2021年）、声源分离（Stoller 等，2018年）或声音事件检测（Lim 等，2017年）。一些作者指出，卷积模型只需少量模型参数，就能达到与循环神经网络（RNN）相近的性能（Takahashi & Mitsufuji，2017年）。另有观点认为，卷积神经网络比循环神经网络（RNN）更容易并行化（Gui 等，2019；Liu 等，2022年；Rybalkin 等，2021年；Kong 等，2021年）。

然而，与可以逐个样本处理输入数据的循环神经网络不同，卷积神经网络需要一块数据才能正常工作。最小数据块大小等于感受野大小，而感受野取决于卷积核大小、步幅、膨胀率以及卷积层的数量。此外，为了减少填充造成的不良边缘效应，可能还需要重叠。因此，标准卷积神经网络模型的延迟高于循环神经网络。

算法延迟与模型的要求和限制有关，例如最小数据块大小（如单个快速傅里叶变换帧的大小）、模型前瞻等，是算法固有的属性。假设所有计算都能瞬时完成，可以将其视为输出信号与输入信号之间的延迟。计算时间是延迟的第二个组成部分。对于卷积神经网络，由于无论所需的输出大小如何，都必须处理整个感受野，因此计算时间并不与数据块大小呈线性关系。

对于视听信号，人类能够察觉到音频与视觉刺激之间超过 10 ms 的延迟（Mcpherson 等人，2016）。不过，对话中可接受的最大延迟可达 40 ms（Staelens 等人，2012；Jaekl 等人，2015年；Ipser 等人，2017年）。为了实现最佳的人机

交互，许多音频应用会将缓冲区大小设为与最大可接受延迟相匹配。

1.1 相关研究

许多研究人员提出了解决方案，以降低信号处理模型的延迟。Wilson 等人（2018年）研究了一种由双向长短期记忆网络（BLSTM）、全连接层（FC）和卷积层构成的模型。首先，他们提出用单向长短期记忆网络替代 BLSTM，并发现这样可以将延迟降低一半，同时对性能影响甚微。其次，他们提出将每个卷积层的感受野改为因果形式，而不是以当前处理的数据点为中心。此外，他们还提出相对于输出对输入特征进行偏移，从而有效提供一定量的未来上下文，作者将其称为前瞻。作者指出，预测未来的频谱图掩码会显著降低准确率：与零前瞻因果模型相比，当输入/输出之间偏移 100 毫秒时，信号失真比（SDR）降低了 6.6 dB。研究者认为，这种影响是因为模型无法立即响应不断变化的噪声和语音特征。

Romaniuk等人（2020）进一步改进了上述模型，移除了长短期记忆网络和全连接层，并将卷积层替换为逐通道卷积后接逐点卷积，同时还对模型进行了其他改动。这些改动使每秒融合乘加运算次数（FMS/s）减少了10倍。然而，计算复杂度的降低伴随着信噪比（SNR）性能下降了10%。作者还引入了对输入短时傅里叶变换帧的部分缓存，并将其称为增量推理。每当新帧到达时，便用最近缓存的输入帧对其进行填充，使其匹配模型的感受野。随后，模型处理组合后的输入，并生成相应的输出帧。

Kondratyuk 等（2021）推出了一个用于视频在线分类的新型卷积神经网络系列（MoViNets）。作者提出了一种更全面的数据缓存方法，即逐层缓存，而非仅输入缓存。MoViNets 将视频分成连续的小片段进行处理，且内存消耗保持恒定。这是通过所谓的流缓冲区实现的，它们会在子片段边界处缓存特征图。使用流缓冲区可使大型三维卷积神经网络的峰值内存消耗最多降低一个数量级。在较小的网络中，内存减少一半所带来的影响则不那么显著。由于该方法面向在线推理，流缓冲区最好与因果网络结合使用。作者通过将右侧（未来）的填充移到左侧（过去）来确保因果性。据报道，流缓冲区会使模型准确率降低约1%，并使计算复杂度略有增加；不过，这可能取决于具体实现。

1.2 新颖性

本文提出了一种用于卷积层的数据缓存新方法，称为短期记忆卷积（STMC）。该方法可处理任意大小的数据块，且不会产生任何额外计算开销（即模型初始化后，处理数据块的计算成本与其大小呈线性关系），从而缩短计算时间。该方法与模型和任务无关，唯一的前提是使用堆叠卷积层。
我们还系统地探讨了算法延迟问题（具体而言即前瞻），讨论转置卷积层的因果性，并提出必要的调整，以确保类自编码器卷积神经网络模型具有因果性。

STMC 层基于以下原则：

- 输入数据以任意大小的数据块在线处理。
- 每个数据块都会经过所有卷积层的传播，并使用移位寄存器缓存每一层的输出（与Romaniuk等人（2020）中的仅输入缓存不同），从而提供所谓的过去上下文。
- 任何卷积层都不会重新计算过去上下文。处理时间序列时，所有计算都只执行一次，与所处理的数据块大小无关。

- 卷积层转换为 STMC 层是在模型训练完成后进行的，因此该转换不会影响训练时间，也不会改变模型输出。

STMC 可视为 Kondratyuk 等（2021）提出的流缓冲技术的一种泛化形式，但它是独立开发的，旨在降低采用标准卷积、带步幅卷积、转置卷积以及池化层的卷积神经网络中的延迟和峰值内存消耗（而 MoViNets 仅采用标准卷积）。STMC 以不同于 MoViNets 的方式处理缓存问题。它使用移位寄存器来缓存所有层的特征图，并根据输入尺寸沿时间轴移动这些特征图。移位寄存器支持处理任意大小的数据块，且数据块大小可以随时间变化。此外，移位寄存器可以缓冲网络的多个状态，从而支持处理步幅和池化层。最后，我们提出了转置 STMC，使其适用于更广泛的网络类型，包括 U-Net。

2 方法

本节介绍短期记忆卷积及其转置对应方法。

2.1 短期记忆卷积

STMC 和转置 STMC（tSTMC）层引入额外的缓存内存，用于存储过去的结果。图 2 展示了我们方法的概念图，其中包含引入的内存与实际卷积之间的交互。缓存内存会将过去计算出的值提供给后续调用的 STMC 层，以便在不进行任何冗余计算的情况下计算最终输出。目标是仅处理每项数据一次。

STMC 移位寄存器的设计基于这样一个观察：在线模型的卷积神经网络层处理的输入数据，与前一步的数据部分重叠。唯一的区别在于，新数据略有偏移，且输入中只有一小部分是全新的。如图 1A 所示，假设有 2 个大小为 2 的卷积核，在时间步 t_1，只有与第 6 帧新数据相关的输出尚未在时间步 t_0 计算出来。为了在不进行任何冗余计算的情况下完整重建输出，卷积结果

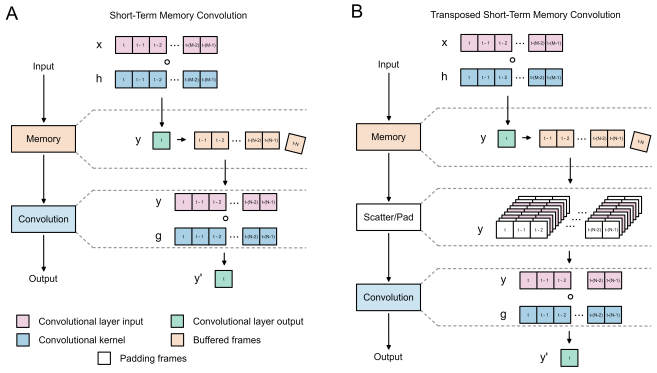
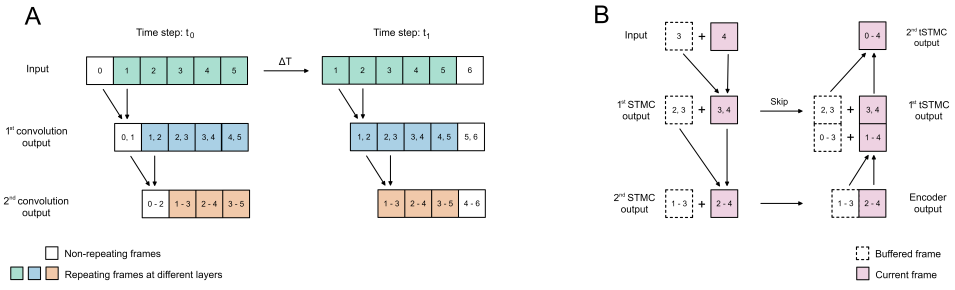


图 2：A）短期记忆卷积。B）转置短期记忆卷积。



Fig图 1：STMC 对网络的影响。A）时域中的卷积冗余效应。B U-带有 STMC/tSTMC 的净网络。)

（特征图）必须保存在网络的每一层（图 1B）。特定 STMC 模块中保存的帧数等于其感受野大小。随后，将先前保存的状态填充到下一卷积层的输入中。该方法能够以高效的方式保留网络中的所有过去上下文。其高效之处在于，所有缓存数据都已经由相应层预处理过。

为便于说明，假设一个由N 个 $X_t \in \mathbb{R}^N$ 样本组成的一维时间序列数据是卷积层在时间 t的输入。该卷积层具有一个卷积核$h_1 \in \mathbb{R}^{M_1}$，在采用 $H_1 \in \mathbb{R}^{N\times(N-M_1+1)}$ 假设使用有效填充的情况下，也可以用托普利兹矩阵表示。我们可以按如下方式计算输出信号 $Y_t = (y_{t-N+M_1},\ y_{t-N+M_1+1},\ \ldots\ldots\ y_t)$：

$$Y_t = \sigma(H_1 \cdot X_t^{\mathrm{T}}) \tag{1}$$

其中，σ 是逐元素应用的激活函数。随后，Y 分别经过第二层（$h_2 \in \mathbb{R}^{M_2}$ 和 $H_2 \in \mathbb{R}^{(N-M_1+1)\times(N-M_1-M_2+2)}$ 处理：

$$Z_t = \sigma(H_2 \cdot Y_t^{\mathrm{T}}) \tag{2}$$

对每个后续时间步 t 执行上述操作的计算复杂度为 O(NM_1) + O(NM_2)。为了降低计算需求，可以采用 Romaniuk 等，2020 中所述的增量推理方案。输入数据将被裁剪为与感受野大小相匹配，在我们的示例中，该大小为 $M_1 + M_2 - 1$。对于大小上限为 M的 K 层，增量推理的计算复杂度为 O(K^2M^2)。

STMC 带来了进一步的改进，因为它缓存的不仅是输入，还包括所有卷积层的全部中间输出。在模型后续调用时，将这些输出与最新数据拼接后使用。该技术确保所有卷积仅处理最新数据，并将卷积核置于最新位置。只需计算最新的帧 y_t =σ($h \cdot X_t$)：

$$Y_t = (Y_{t-1} \mid \sigma(h_1 \cdot X_t^{\mathrm{T}})) \tag{3}$$

其中，$X_t \in \mathbb{R}^{M_1}$ 是最新的输入帧，而 | 表示移位寄存器执行的操作。对于任意序列 P和元素 α，有：

$$P = (p_1, p_2, ..., p_n) \tag{4}$$

$$(P|\alpha) = (p_2, p_3, ..., p_n, \alpha) \tag{5}$$

因此，Z_t 的计算方式为：

$$Z_t = (Z_{t-1}|\sigma(h_2 \cdot (Y_{t-1}|\sigma(h_1 \cdot X_t^{\mathrm{T}})) \tag{6}$$

在上述假设下，该方案的复杂度为 O(M_1) + O(M_2)；对于多层网络，则为 O(KM)。

值得注意的是，STMC 的效果会随着网络深度的增加而增强。缓冲区Y_{t-1} 可以用大小为 $M_1 - 1$ 的简单移位寄存器实现（图 2）。实际应用中，该方法可能会因 CNN 架构不同而更加复杂。例如，步幅卷积需要缓冲多个状态（使用多个移位寄存器）。不过，在这种情况下，STMC 能通过缓解步幅导致的感受野增大，大幅提升模型的处理速度，因为步幅卷积会使感受野扩大步幅倍。相比之下，增量推理会受到步幅卷积的显著影响，因为更大的感受野需要上层进行更多计算。

STMC 也可以应用于转置卷积。一种方法是将 STMC 与带零填充的标准卷积结合使用，以便进行上采样。不过，还有一种优化程度更高的方法

存在一种方法，我们将其称为**转置 *STMC***（tSTMC）。图 3 展示了我们对转置卷积的实现示意图，该实现可以利用缓存，并呈现每次操作的部分输出。值得注意的是，所有卷积前算子仅应用于频率轴。首先，通过引入输入内部的零值区域，应用散射来扩展输入。每个区域的大小取决于前一个转置卷积的步幅。其次，添加填充以确保所需的输出形状，然后应用卷积。

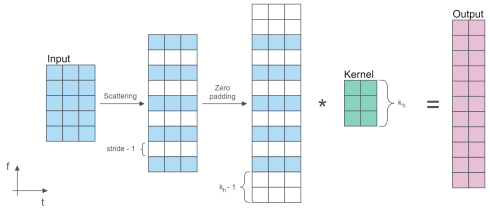


图 3：转置短期记忆卷积。

2.2 局限性

STMC 是一种与任务无关的方法，所提出的缓存和推理方案可以应用于任何任务，而不受领域限制。此外，STMC 也可以被视为与模型无关，因为无论模型中其他模块如何，它都可以应用于模型任意位置的堆叠卷积层。

如前所述，STMC 的第一个局限性在于，它只能应用于卷积层和池化层。不同于仅将最近数据作为输入的循环神经网络和长短期记忆网络，卷积模型需要更长的数据片段作为输入。STMC 只需缓存输入片段中重叠的部分，以及连续模型调用之间网络内的所有中间结果。

第二个局限性是，STMC 的具体实现取决于模型的架构。本文介绍了如何为 U-Net 和 GhostNet 模型实现 STMC。虽然可以为具有步幅和膨胀率的层实现 STMC，但这会显著增加网络的复杂度。例如，步幅卷积需要每层使用多个移位寄存器。

最后，STMC 应与因果卷积结合使用。由于缺乏未来上下文，因果卷积通常比标准卷积具有更低的延迟。此外，在转置卷积中，因果卷积实际上是“反因果的”，因为反卷积输出仅取决于当前和未来的输入值（与因果情况相反，后者的输出仅取决于过去和当前的输入）。通过相对于模型输出移动模型输入，可以抵消这一影响；对于堆叠卷积层，这种方法有效，但在更复杂的架构中可能需要进行一些额外修改。

3 实验

本节介绍用于评估所提出方法推理时间和内存消耗的实验。我们将介绍模型架构、实验任务、所选数据集以及训练设置，并说明选择这些方案的理由。

3.1 语音分离

3.1.1 实验任务

我们选择语音分离作为主要任务，以展示 STMC/tSTMC 方案带来的收益。我们对这一问题采用广义定义：从背景干扰中提取通常意义上的语音信号。由于我们的重点是获取干净的语音信号，因此该任务也可理解为语音去噪或噪声抑制。为清晰起见，我们将所述任务称为语音分离。

桌面端评估 我们采用带有 C++binary 的 TFLite 模型基准测试工具，评估推理时间和峰值内存消耗。每个模型均在 Intel(R) Xeon(R) Gold 6246R CPU 的 8 个核心上独立运行 1,000 次，该处理器的时钟频率为 3.40 GHz；随后计算平均推理时间和平均峰值内存消耗。此外，我们还计算了平均尺度不变信噪比改进指标（SI-SNRi），以确保所提出的优化方法不会降低输出质量。

延迟 L使用以下公式计算：

$$L = l_c + l_f + t_i + t_{\text{STFT}} + t_{\text{iSTFT}} \tag{7}$$

组件 l_c 和 l_f 构成算法延迟： l_c 是当前音频块的长度， l_f是模型前瞻信号的长度——二者均以时间为单位。组件 t_i、t_{STFT} 和 t_{iSTFT} 与计算成本有关： t_i 是推理时间，另外两项分别是计算短时傅里叶变换及其逆变换所花费的时间。请注意，对于因果模型，未来音频 l_f 的长度为零。每个步骤中，短时傅里叶变换和逆短时傅里叶变换都只针对一帧进行计算。我们的研究表明，借助 STMC， t_i 有所改善；此外，由于引入了因果性， l_f 也有所改善。对于每个模型，延迟均通过能够实现实时处理（即实时因子 (RTF) 小于或等于 1）的最短输入长度进行估算。

在移动设备上的评估 为确认所提方法能够移植到其他设备，我们还在移动设备上采用与桌面端相同的设置对模型进行了评估。我们使用了搭载 Qualcomm Snapdragon 888 的 Samsung Fold 3 SM-F925U。模型通过 TFLite 模型基准测试工具进行分析，不过这次使用了 4 个核心。

不同模型大小下的 STMC 为了直观了解 STMC 在不同大小的模型上的表现，我们将 STMC 应用于卷积核数量经过缩放的同一 U-Net 架构。我们测试了 16 个模型，其大小从 76 KB 到 32 806 KB 不等。为平滑测得的特征，每种模型大小都训练了 5 次，所有结果均报告平均值。每个模型的 TFLite 转换方式与主要实验类似。

前瞻的影响 由于并非所有应用都要求如此低的延迟，仍可通过引入 STMC 来降低计算复杂度。我们提供了实验结果，展示使用不同前瞻长度的模型的尺度不变信噪比改进。每种前瞻长度均训练了 5 个模型。测试的前瞻范围为 0 至 7 帧，步长为 1 帧；其中 0 对应因果网络，7 对应主要实验中的非因果网络。

3.1.2 模型与训练过程

本研究基于 U-Net 架构，该架构由卷积、转置卷积以及跳跃连接组成。因此，正如 A 节所述，由于解码器具有“反因果性”，这给 STMC 带来了更大的挑战。此外，U-Net 广泛用于解决音频领域任务，并已证明能够取得很好的效果。总共比较了七种 U-Net：U-Net、U-Net $_{\text{INC}}$、U-Net $_{\text{STMC}}$、因果 U-Net、因果 U-Net $_{\text{INC}}$、因果 U-Net $_{\text{STMC}}$ 和因果 U-Net $_{\text{tSTMC}}$。这些模型在因果性以及 STMC 应用于编码器和解码器方面有所不同。此外，U-Net 和因果 U-Net 都在一次推理中处理全部数据，因此实际上属于离线模型。其他模型则以在线模式运行，分块处理数据。

U-Net 作为基准模型，我们采用了一个基于 U-Net 架构（Ronneberger 等，2015）的网络。我们通过移除最大池化操作简化了原始架构，因此编码完全由卷积完成。我们针对音频分离任务调整了该模型，并在深度为 7 的模型上取得了令人满意的结果。每个卷积层包含若干个长度为 3 的卷积核，作用于时域，因此组装后的网络感受野为 29。频域中的卷积核大小设为 5。由于基准网络由以当前时间帧为中心的卷积层组成（Favero 等，2021），因此可以得出，过去和未来上下文的感受野均为 14。网络采用实部和虚部分别作为独立通道的短时傅里叶变换输入，并以复数掩码作为输入信号的网络输出。短时傅里叶变换窗口大小为 1024，跳步为 256。

因果 U-Net 为了展示因果性的影响，我们还训练了因果 U-Net 模型。在所有考虑的因果模型中，我们在时域中使用大小为 2 的卷积核，以降低计算成本和感受野，并压缩模型大小。初步测试表明，这种改变不会影响因果模型的输出质量。这种方法既能保留必要的计算，又能去除额外的填充操作。网络的感受野实际上缩小了一半。为实现因果性，我们将模型输出移位 7 个 STFT 帧。为了对齐网络内部的数据，所有跳跃连接也需要根据其在模型中的深度进行相应移位。

增量推理 我们将增量推理方法（Romaniuk 等，2020）应用于 U-Net 和因果 U-Net：将原始输入尺寸裁剪至感受野大小，从而为 U-Net 生成包含 29 个 STFT 帧的输入，为因果 U-Net 生成包含 15 个 STFT 帧的输入。我们将该网络称为 U-Net $_{\text{INC}}$ 和因果 U-Net $_{\text{INC}}$。

短期记忆卷积 与增量推理类似，STMC 同样应用于 U-Net 和因果 U-Net 网络。首先，我们在编码器和解码器内部应用 STMC，并采用零填充和输出裁剪（即不使用 tSTMC）。我们将这些网络称为 U-Net $_{\text{STMC}}$ 和因果 U-Net $_{\text{STMC}}$。为展示采用裁剪的 STMC 与采用 tSTMC 的 STMC 在推理时间上的差异，我们还单独训练了一个同时包含 STMC 和 tSTMC 的因果模型。该网络记为因果 U-Net $_{\text{tSTMC}}$。对于这种方法，U-Net 和因果 U-Net 在时间轴上的输入尺寸分别设为 3 和 2。

数据集和训练条件 训练和验证网络时采用了深度噪声抑制（DNS）挑战赛 - Interspeech 2020 数据集（Reddy 等人，2020），该数据集采用 CC-BY 4.0 许可，且易于处理，适用于语音分离任务。训练过程中，模型接触了约 1 亿个随机切分的 3 秒音频样本。所有模型（U-Net、因果 U-Net 和因果 U-Net$_{tSTMC}$）均在 Nvidia A100 上训练，批量大小为 128，使用 Adam 优化器，学习率为 1e-3。每个模型训练 100 个 epoch，每个模型耗时约 90 小时。由于训练方案和所用损失函数的更多细节既不会影响所提方法的实用性，也不会影响其有效性，因此在此略去。

所有模型均使用转换器的默认设置转换为 TFLite 格式，且未进行量化。

3.2 声学场景分类

3.2.1 实验任务

为了展示我们的方法对小型卷积神经网络模型同样有效，我们将其应用于经过声学场景分类（ASC）任务训练的 GhostNet（Han 等，2020）架构。我们在中央处理器（Intel(R) Xeon(R) Gold 6246R CPU，主频 3.40 GHz）、图形处理器（Nvidia A100）以及启用 XNNPACK 优化的中央处理器上测试了推理时间和峰值内存消耗。作为对比，我们还测试了长短期记忆网络和 MobileViT（Mehta 和 Rastegari，2022）网络。

3.2.2 模型与训练过程

GhostNet 我们的分类网络由 3 个 Ghost 瓶颈模块组成，并以一个卷积层和平均池化层结束。模型的输入为幅度谱图，其时间轴大小为 94，频率轴大小为 256。

长短期记忆网络 我们测试了循环神经网络架构，其中包含 2 个长短期记忆网络模块，单元数分别为 64 和 32，以及 2 个全连接层，单元数分别为 16 和 1。模型接受一个时间帧的幅度谱图作为输入，其中包含 256 个频率箱。

MobileViT 我们还实现并训练了 MobileViT 网络。与 (Mehta & Rastegari, 2022) 中描述的原始模型不同，我们将输入通道数设为 1，输出类别数也设为 1。

数据集和训练条件 我们采用 TAU Urban Acoustic Scene 2020 Mobile 数据集 (Heit- tola et al., 2020)，作为声学场景分类任务中网络训练和验证所用的数据集。我们按照指定顺序，通过峰值归一化、裁剪以及转换为幅度谱图来扩充数据。我们遵循 DCASE 2020 挑战赛任务 1A 关于训练集/测试集划分的说明。由于训练方案和所用损失函数的更多细节既不影响所提方法的实用性，也不影响其有效性，因此未予赘述。
所有模型均采用转换器的默认设置转换为 TFLite 格式，且未进行量化。

4 结果

4.1 语音分离

桌面端结果表1并列展示了所有模型的结果。与标准卷积相比，增量推理（INC）方法在非因果和因果设置下分别实现了 8.00 倍和 29.74 倍的延迟降低。我们的方法（STMC）优于 INC，在非因果变体下进一步降低了 1.32 倍延迟，在因果变体下则降低了 2.52 倍。STMC 还使推理时间降低了约 5.68 倍

表1：使用 TFLite 模型基准测试工具在桌面端进行评估的结果。“C.”表示因果。*评估指标的差异是由于模型分别训练所致。

Name	尺度不变信噪比改进 (dB)	推理时间（毫秒）	延迟 y (ms)	峰值内存 y (MB)	Size (KB)
U-Net	**8.00**	1381.02 ± 17.62	4403.22	170.36	14 025
INC	**8.00**	151.92 ± 12.25	550.54	48.97	14 025
STMC	**8.00**	86.33 ± 4.33	416.03	51.31	14 035
C. U-Net	7.58	962.61 ± 25.91	3984.81	138.18	**9 366**
C. INC	7.58	55.64 ± 2.47	133.98	31.57	9 367
C. STMC	7.58	10.37 ± 0.69	52.07	27.50	9 376
C. tSTMC	7.44*	**9.80 ± 0.97**	**51.50**	**22.20**	9 373

（与 INC 相比）。STMC 对非因果 U-Net 的影响较小，因为 U-Net$_{\text{STMC}}$ 的大部分计算成本来自等待未来上下文的需求（相对于输出居中的卷积核）。在因果 U-Net 模型的解码器中实现转置 STMC，仅略微改善了延迟（0.57 毫秒），但将峰值内存消耗降低了约 20%。标准、INC 和 STMC 变体的模型大小相近。因果模型比非因果模型小约 33%，这是由于它们使用的卷积核较小，详见第 3.1.2 节。

表 2：使用 TFLite Model Benchmark Tool 在移动设备上进行的评估结果。“C.”表示因果。

Name	推理時間 (毫秒)	延迟 (ms)	峰值内存 (MB)
U-Net	262.44 ± 26.81	3 284.64	164.93
INC	38.51 ± 4.34	323.72	44.14
STMC	19.26 ± 1.02	288.96	46.59
C. U-Net	240.95 ± 30.4	3 263.15	133.60
C. INC	14.02 ± 0.71	50.74	26.53
C. STMC	3.15 ± 0.10	28.85	21.54
C. tSTMC	**2.77 ± 0.26**	**28.47**	**16.11**

移动设备上的结果 移动设备评估的结果汇总于表 2 中。它们进一步证实了从桌面端结果中得出的结论。由于本次测试所使用的模型与桌面端评估中的模型相同，因此表中未列出模型大小和尺度不变信噪比改进评估指标。

尽管两个平台采用了相同的评估方法，但移动设备上的推理时间和延迟仍显著低于桌面端。因果 U-Net$_{\text{INC}}$ 与因果 U-Net $_{\text{tSTMC}}$ 之间的推理时间缩减倍数为 5.06。在该设备上使用 tSTMC 的影响相对桌面端更大，因为其将推理时间缩短了 12.06%（相比之下为 5.50%）。与桌面端类似，因果 U-Net$_{\text{tSTMC}}$ 实现了最低的峰值内存消耗。

不同模型大小下的 STMC 这组实验的结果如图 4 所示，由 4 张图组成。如前所述，INC 和 STMC 模型的尺度不变信噪比改进评估指标相同，该指标与模型大小的关系见图 4A。STMC 带来的绝对峰值内存消耗改善随模型大小线性增长（图 4B），但相较于 INC 的相对改善则从 16% 线性下降至 11%。在所有测试模型中，STMC 都显著影响推理时间，如图 4C-D 所示。对于两种推理模式，与绝对峰值内存改善类似，推理时间似乎也随模型大小线性增长（图 4C）。对于这一架构，STMC 相较于 INC 的加速比保持恒定，约为 81%；但模型大小为 1.32 MB 及以下时，加速比会降至约 74%。

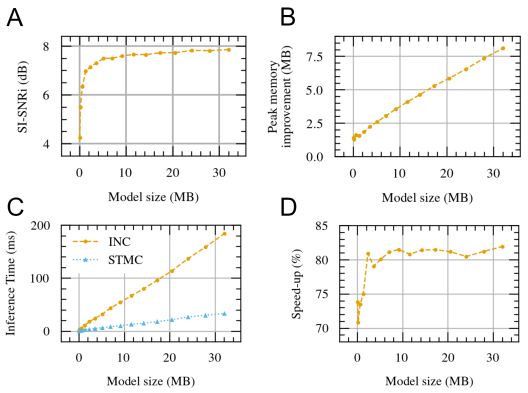


图 4：模型大小对 STMC 的影响。A）尺度不变信噪比改进。B）峰值内存消耗的改善。C）推理时间。D）STMC 相对于 INC 的加速比。

前瞻的影响 如图 5 所示，前瞻对评估指标有积极影响。还应注意的是，该实验确认评估指标下降 6% 完全归因于缺少未来内容，因为在本次测试中，我们实际上对主实验中的因果网络和非因果网络进行了插值。所确定的特征并非线性特征，最初的帧似乎比最右侧、距离更远的未来帧更为重要，因为第一帧使尺度不变信噪比改进提高了 0.177 dB，而最后一帧仅提高了 0.019 dB。在此图中，我们加入了误差棒，因为同一模型之间的差异比本文介绍的其他任何实验都更为显著。在特定前瞻对应的误差棒范围内，包含了我们为每种特定前瞻训练的全部 5 个模型在验证数据集上取得的所有尺度不变信噪比改进结果。

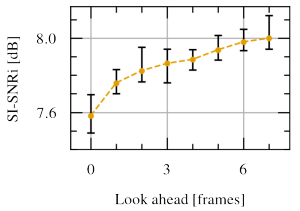


图 5：前瞻对尺度不变信噪比改进的影响。

4.2 声学场景分类

将 STMC 应用于 Ghost-Net 架构不会对准确率指标产生影响。此外，我们观察到中央处理器推理时间减少了 4.84 倍，图形处理器推理时间改善了 34%，XNNPACK 的推理时间减少了 2.24 倍。在峰值内存消耗方面，我们观察到中央处理器的峰值内存减少了 22%，XNNPACK 的峰值内存减少了 30%，而模型大小增加了 4 KB。我们不报告图形处理器推理的峰值内存消耗，因为

表 3：声学场景分类实验结果。

网络	Top-1 准确率（%）	推理时间（毫秒）	峰值内存 (MB)	数量参数	输入尺寸
GhostNet	65.83	4.706 ± 6.167	9.76	8 296	94×256×1
GhostNet GPU		0.912 ± 0.504	—		
GhostNet XNNPACK		1.086 ± 5.612	11.49		
GhostNet STMC 循环处理器	65.83	0.973 ± 0.866	7.59	8 300	1×256×1
GhostNet		0.602 ± 0.321	—		
GhostNet STMC XNNPACK		0.484 ± 2.447	8.10		
LSTM	60.84	0.095 ± 0.010	7.50	95 137	1×256
LSTM GPU		0.099 ± 0.074	—		
长短期记忆网络 XNNPACK		0.119 ± 0.348	7.70		
MobileViT XXS	69.58	52.948 ± 11.456	38.11	1 306 049	256×256×1
MobileViT XXS GPU		419.203 ± 90.989	—		

因为我们使用的工具不提供这些信息。通过将 STMC GhostNet 与 LSTM 进行比较，我们可以观察到，LSTM 的推理时间最多快 10 倍，但代价是参数数量增加了 12 倍。需要注意的是，在使用 XNNPACK 时，LSTM 相较于 STMC 在推理时间上的优势降至快 4 倍。此外，我们的测试表明，就推理时间而言，卷积神经网络总体上比长短期记忆网络更加不稳定。LSTM 模型的准确率为 60.84%，低于 STMC GhostNet 的 65.83% 准确率。在参数数量为 130 万的情况下，MobileViT 取得了观测到的最高准确率 69.58%，同时推理时间也是最长的，为 52 毫秒。该模型在给定的声学场景分类任务中取得了最佳评估指标，但较长的推理时间使其不适合在线处理。在 TAU 数据集上表现最佳、并提交至 DCASE 2020 挑战赛任务 1A 的模型准确率为 74.4%，因此，我们研究中的所有模型都可以被视为可行方案。

5 结论

本文介绍了一种降低推理时间、延迟和内存消耗的方法。该方法依靠在每次迭代中缓存层输出，目标是只处理所有数据一次，并重复利用所有卷积层的历史结果。我们在两种设备（台式计算机和手机）上测试了该方法的语音分离任务，以及在 x86 CPU 和 A100 GPU 上使用 Ghost-Net 的声学场景分类任务。音频分离方面，我们总共比较了 U-Net 架构的 7 个变体，它们在因果性和缓冲类型方面各不相同。在 x86 和 ARM 上，与增量推理相比，推理时间均缩短了约 5 倍。增量推理被认为是使用卷积模型进行在线处理的最先进方法。STMC方法的延迟也降低了 44%–61%，峰值内存消耗降低了 30%–39%，具体幅度取决于测试平台。测试中使用的所有因果模型，其尺度不变信噪比改进均比对应的非因果模型低约 6%。分数下降的原因被归结为缺少未来上下文，而非所使用的缓存方法。

STMC方法已被证明能够显著减少卷积神经网络中的运算次数，这对于许多深度学习应用可能至关重要，例如在嵌入式设备上部署模型。该方法可应用于现有的大多数卷积神经网络架构，并可与时间序列处理所采用的多种不同方法相结合，例如扩张因果卷积（van den Oord等，2016）或时序卷积（Lea等，2017年）。使用STMC及其转置对应方法——tSTMC——无需重新训练模型。不过，为了实现接近实时的处理，强烈建议使用因果模型。

本文所介绍的方法也适用于本研究所呈现的音频分离任务以外的应用。由于 STMC 方法能够缩短推理时间和延迟，并减少峰值内存消耗，因此可用于数据量超出可用内存或计算能力所能处理范围的系统，例如：医学影像（Blumberg 等，2018年）、射电天文成像（Corda 等，2020年）、现场可编程门阵列（Ma 等，2018年）、视频分析（Lea 等，2017年）和校验和计算（Filippas 等，2022年）。我们还认为，所介绍的方法可能适用于面向在线处理应用的低延迟片上系统（SoC）（Park 等，2022年）。

参考文献

Stefano B. Blumberg、Ryutaro Tanno、Iasonas Kokkinos 和 Daniel C. Alexander。《更深层次的图像质量迁移：为三维图像训练低内存神经网络》。载于医学图像计算与计算机辅助干预——*MICCAI 2018*，第 118–125 页。Springer International Publishing，2018年。ISBN 978-3-030-00928-1。

Stefano Corda、Bram Veenboer、Ahsan Javed Awan、Akash Kumar、Roel Jordans 和 Henk Cor- poraal。高分辨率射电天文成像的近内存加速。载于 *2020*年第九届地中海嵌入式计算会议（*MECO*），第 1–6 页，2020年。doi：10.1109/ MECO49872.2020.9134089。

Alessandro Favero、Francesco Cagnetta 和 Matthieu Wyart。局部性在卷积教师-学生场景中克服维数灾难。载于神经信息处理系统进展，2021年。URL：https://openreview.net/forum?id=sBBnfOFtPc。

Dionysios Filippas、Nikolaos Margomenos、Nikolaos Mitianoudis、Chrysostomos Nicopoulos 和 Giorgos Dimitrakopoulos。低成本在线卷积校验和检查器。*IEEE* 超大规模集成电路系统汇刊，30(2)：201–212，2022年。doi：10.1109/TVLSI.2021. 3119511。

Tao Gui、Ruotian Ma、Qi Zhang、Lujun Zhao、Yu-Gang Jiang 和 Xuanjing Huang。基于词典重新思考的中文命名实体识别卷积神经网络。载于第二十八届国际人工智能联合会议论文集，*IJCAI-19*，第 4982–4988 页。国际人工智能联合会议组织，2019 年 7 月。doi：10.24963/ijcai.2019/692。

Kai Han、Yunhe Wang、Qi Tian、Jianyuan Guo、Chunjing Xu 和 Chang Xu。GhostNet：通过廉价操作获得更多特征。载于*IEEE/CVF*计算机视觉与模式识别会议（*CVPR*）论文集，2020 年 6 月。

Toni Heittola、Annamaria Mesaros 和 Tuomas Virtanen。TAU Urban Acoustic Scenes 2020 Mobile，开发数据集，2020 年 2 月。网址：https://doi.org/10.5281/zenodo. 3819968。

Alberta Ipser、Vlera Agolli、Anisa Bajraktari、Fatimah Al-Alawi、Nurfitriani Djaafara 和 Elliot D. Freeman。视觉与听觉持续不同步：通过唇声整合的内隐测量揭示视听同步中的稳定个体差异。*Scientific Reports*，7(1): 46413，2017年4月。ISSN 2045-2322。doi: 10.1038/srep46413。

Philip Jaekl、Jakob Seidlitz、Laurence R. Harris 和 Duje Tadin。视听延迟作为视觉距离的一种新线索。*PLOS ONE*，10(10):1–12，2015年10月。doi: 10.1371/journal.pone.0141125。

Dan Kondratyuk、Liangzhe Yuan、Yandong Li、Li Zhang、Mingxing Tan、Matthew Brown 和 Boqing Gong。MoViNets：用于高效视频识别的移动视频网络。载于*IEEE/CVF*计算机视觉与模式识别会议（*CVPR*）论文集，第16020–16030页，2021年6月。URL https://openaccess.thecvf.com/content/CVPR2021/ html/Kondratyuk_MoViNets_ Mobile_Video_Networks_for_Efficient_Video_Recognition_CVPR_2021_paper.html。

Jun Kong、Leixin Zhang、Min Jiang 和 Tianshan Liu。将多层级卷积神经网络和注意力机制融入中文临床命名实体识别。生物医学信息学杂志，116:103737，2021年。ISSN 1532-0464。doi: 10.1016/j.jbi.2021.103737。

Colin Lea、Michael D. Flynn、Ren´e Vidal、Austin Reiter 和 Gregory D. Hager。用于动作分割和检测的时间卷积网络。载于第*30*届*IEEE*计算机视觉与模式识别会议，*CVPR 2017*，第1003–1012页。电气电子工程师学会有限公司，2017年11月。doi: 10.1109/CVPR.2017.113。

Hyungui Lim、Jeongsoo Park 和 Yoonchang Han。使用一维卷积循环神经网络进行罕见声音事件检测。技术报告，DCASE 2017挑战赛，2017年9月。

Yingying Liu、Peipei Li 和 Xuegang Hu。结合上下文相关特征与多阶段注意力网络进行短文本分类。计算机语音与语言，71:101268，2022年。ISSN0885-2308。doi：10.1016/j.csl.2021.101268。

Yufei Ma、Yu Cao、Sarma Vrudhula 和 Jae-sun Seo。优化卷积运算，以加速现场可编程门阵列上的深度神经网络。*IEEE* 超大规模集成电路系统汇刊，26(7):1354–1367，2018年。doi：10.1109/TVLSI.2018.2815603。

Andrew Mcpherson、Robert Jack 和 Giulio Moro。动作-声音延迟：我们的工具够快吗？发表于国际音乐表达新界面会议，2016年4月。

Sachin Mehta 和 Mohammad Rastegari。Mobilevit：轻量级、通用且适用于移动设备的视觉Transformer。发表于国际学习表征会议，2022年。网址：https://openreview.net/forum?id=vh-0sUt8HlG。

Shervin Minaee、Yuri Y. Boykov、Fatih Porikli、Antonio J Plaza、Nasser Kehtarnavaz 和 Demetri Terzopoulos。基于深度学习的图像分割：综述。*IEEE* 模式分析与机器智能汇刊，第 1–1 页，2021年。doi：10.1109/TPAMI.2021.3059968。

Jun-Seok Park、Changsoo Park、Suknam Kwon、Hyeong-Seok Kim、Taeho Jeon、Yesung Kang、Heonsoo Lee、Dongwoo Lee、James Kim、YoungJong Lee、Sangkyu Park、Jun-Woo Jang、SangHyuck Ha、MinSeong Kim、Jihoon Bang、Suk Hwan Lim 和 Inyup Kang。一款面向硬件利用率的多模式 8k-MAC 神经处理单元，采用统一多精度数据通路，基于 4nm 旗舰移动SoC。发表于*2022*年*IEEE*国际固态电路会议（*ISSCC*），第 65 卷，第 246–248 页，2022年。doi：10.1109/ISSCC42614.2022.9731639。

Waseem Rawat 和 Zenghui Wang。用于图像分类的深度卷积神经网络：全面综述。*NeuralComputation*，29(9):2352–2449，2017年。doi：10.1162/neco a 00990。

Chandan KA Reddy、Vishak Gopal、Ross Cutler、Ebrahim Beyrami、Roger Cheng、Harishchandra Dubey、Sergiy Matusevych、Robert Aichner、Ashkan Aazami、Sebastian Braun 等。Interspeech 2020 深度噪声抑制挑战赛：数据集、主观测试框架和挑战赛结果。载于*Interspeech 2020*会议论文集，2020年。

Michal Romaniuk、Piotr Masztalski、Karol Piaskowski 和 Mateusz Matuszewski。利用移动音频流网络实现高效的低延迟语音增强。载于*Interspeech 2020*会议论文集，第 3296–3300 页，2020年。doi：10.21437/Interspeech.2020-2443。

Olaf Ronneberger、Philipp Fischer 和 Thomas Brox。U-net：用于生物医学图像分割的卷积网络。发表于*MICCAI*，2015年。

Vladimir Rybalkin、Jonas Ney、Menbere Kina Tekleyohannes 和 Norbert Wehn。当大规模 GPU 并行性仍然不够时：一种新型二维LSTM神经网络硬件架构。*ACM* 可重构技术系统汇刊，15(1)，2021年11月。ISSN 1936-7406。doi：10.1145/ 3469661。

Nicolas Staelens、Jonas De Meulenaere、Lizzy Bleumers、Glenn Van Wallendael、Jan De Cock、Koen Geeraert、Nick Vercammen、Wendy Van den Broeck、Brecht Vermeulen、Rik Van de Walle 和 Piet Demeester。评估音视频同步对于视频序列同声翻译的重要性。多媒体系统，18(6)：445–457，2012年11月。ISSN 1432-1882。doi：10.1007/s00530-012-0262-4。

Daniel Stoller、Sebastian Ewert 和 Simon Dixon。Wave-u-net：一种用于端到端音源分离的多尺度神经网络。载于第*19*届国际音乐信息检索学会会议，2018年。网址：http://ismir2018.ircam.fr/doc/pdfs/205_Paper.pdf。

YuhangSun、Linju Yang、HuifengZhu 和JieHao。用于相位感知语音增强的漏斗式深度复数 U-Net。载于*Interspeech 2021* 论文集，第161–165页，2021年。doi：10.21437/Interspeech. 2021-10。

Naoya Takahashi 和 Yuki Mitsufuji。用于音源分离的多尺度多频带密集网络。发表于*2017*年 *IEEE* 音频与声学信号处理应用研讨会 *(WASPAA)*，第 21–25 页，2017年。doi: 10.1109/WASPAA.2017.8169987。

A¨aron van den Oord、Sander Dieleman、Heiga Zen、Karen Simonyan、Oriol Vinyals、Alex Graves、Nal Kalchbrenner、Andrew Senior 和 Koray Kavukcuoglu。《WaveNet：原始音频生成模型》。发表于 *ISCA* 第九届语音合成研讨会 *(SSW 9)*，第 125 页，2016。网址：https://arxiv.org/pdf/1609.03499.pdf。

A¨aron Van Den Oord、Nal Kalchbrenner 和 Koray Kavukcuoglu。Pixel recurrent neural networks。载于第 *33* 届国际机器学习大会论文集（第 *48* 卷），ICML'16，第 1747–1756 页。JMLR.org，2016。网址：http: //proceedings.mlr.press/v48/oord16.pdf。

Kevin Wilson、Michael Chinen、Jeremy Thorpe、Brian Patton、John Hershey、Rif A. Saurous、Jan Skoglund 和 Richard F. Lyon。探索低延迟语音增强模型的权衡。载于第 *16* 届国际声信号增强研讨会论文集，2018年。

Zhong-Qiu Zhao、PengZheng、Shou-TaoXu 和 Xindong Wu。深度学习目标检测：综述。*IEEE* 神经网络与学习系统汇刊，30(11):3212–3232，2019。doi: 10.1109/TNNLS.2018.2876865。

高效的因果性

在线推理中可以使用非因果卷积，方法是等待足够长的时间，直到感受野被“未来”数据填满，但这会增加延迟；或者用零值或其他值（例如对称地反射数据）替代未来数据。然而，后一种方法会降低最新帧的输出质量，而这些帧恰好是在线推理中最重要的。因此，使用卷积网络进行稳健的在线推理通常需要因果性。STMC 也需要因果性，因为如第 2.1 节所述，STMC 层会缓存过去上下文。

严格来说，卷积运算本身并不对因果性作出任何假设。在许多应用中，尤其是在计算机视觉中，卷积核会以当前关注的输出元素为中心。例如，设 C 是一个卷积层，其卷积核的维度为 3×1，并采用有效填充。设 A 和 B 分别是样本输入张量和输出张量，其维度分别为 5×1 和 3×1。于是有：

$$CA^T = \begin{bmatrix} k_1 & k_2 & k_3 & 0 & 0 \\ 0 & k_1 & k_2 & k_3 & 0 \\ 0 & 0 & k_1 & k_2 & k_3 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{bmatrix} = \begin{bmatrix} b_2 \\ b_3 \\ b_4 \end{bmatrix} = B^T \tag{8}$$

以这种方式使用卷积时，需要过去和未来的上下文。

实现因果性的一种方法是使用掩码卷积，即在与输入进行卷积之前，先将卷积核与非对称掩码相乘（Van Den Oord 等，2016）。然而，由于需要额外的乘法运算，掩码卷积比标准卷积更耗费资源。因此，在信号处理中，更理想的做法是通过相对于输出移动输入来实现因果性（van den Oord 等，2016）。换言之，可以对张量 B 的元素重新编索引，以体现时间依赖，使得对于 $j > i$，b_i 不依赖于任何 a_j：

$$CA^T = \begin{bmatrix} k_1 & k_2 & k_3 & 0 & 0 \\ 0 & k_1 & k_2 & k_3 & 0 \\ 0 & 0 & k_1 & k_2 & k_3 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{bmatrix} = \begin{bmatrix} b_3 \\ b_4 \\ b_5 \end{bmatrix} = B^T \tag{9}$$

$$b_i = k_1 a_{i-2} + k_2 a_{i-1} + k_3 a_i \tag{10}$$

我们的案例研究还要求模型从卷积表示中重建任务指定版本的输入信号。获得卷积逆运算最简单且有效的方法，是对其托普利兹矩阵执行转置类运算，如公式（11）所示。请注意，卷积和转置卷积不共享卷积核参数，但它们的矩阵在形状和结构上互为转置关系。

An 当“因果”卷积被转置时，会出现一个有趣的现象，因为这时我们会得到 t:

$$C'B^T = \begin{bmatrix} k'_1 & 0 & 0 \\ k'_2 & k'_1 & 0 \\ k'_3 & k'_2 & k'_1 \\ 0 & k'_3 & k'_2 \\ 0 & 0 & k'_3 \end{bmatrix} \begin{bmatrix} b_3 \\ b_4 \\ b_5 \end{bmatrix} = \begin{bmatrix} a'_1 \\ a'_2 \\ a'_3 \\ a'_4 \\ a'_5 \end{bmatrix} = A'^T \tag{11}$$

$$a'_i = k'_3 b_i + k'_2 b_{i+1} + k'_1 b_{i+2} \tag{12}$$

这实际上是“反因果”的，即反卷积输出仅取决于当前和未来的输入值（与因果情况相反，因果情况下输出仅取决于过去和当前的输入值）。可以通过相对于模型输出来平移模型输入，以抵消这一影响：

$$C'B^T = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ k'_1 & 0 & 0 \\ k'_2 & k'_1 & 0 \\ k'_3 & k'_2 & k'_1 \end{bmatrix} \begin{bmatrix} b_3 \\ b_4 \\ b_5 \end{bmatrix} = \begin{bmatrix} a'_1 \\ a'_2 \\ a'_3 \\ a'_4 \\ a'_5 \end{bmatrix} = A'^T \tag{13}$$

$$a'_i = k'_3 b_{i-2} + k'_2 b_{i-1} + k'_1 b_i \tag{14}$$

这种方法适用于堆叠卷积层，但对于更复杂的架构，可能需要进行一些修改。例如，U-Net 模型在卷积和转置卷积之间设有跳跃连接。每条跳跃连接传递的数据也必须相对于转置卷积的输出进行适当平移。平移量取决于网络架构，在不同层级可能有所不同。