MS-TCN++: 用于动作分割的多阶段时域卷积

李仕杰 *, Yazan Abu Farha*, 刘云, 程明明, Juergen Gall

摘要—随着深度学习在短的剪辑视频分类中的成功应用，对长的、未经剪辑的视频的时域分割和活动分类受到了越来越多的关注。动作分割中，最先进的方法是利用多层时域卷积和时域池化实现的。尽管这些方法有捕捉时域相关性的能力，他们的预测存在过度分割的错误。在本文中，我们提出了一个解决时间动作分割任务的多阶段架构，克服了以往方法的局限性。第一阶段产生一个初步的预测，下一个阶段会对其进行改良。在每一阶段，我们堆叠几层扩展的时域卷积，它覆盖着一个参数很少的大接受域。虽然这种架构已经表现得很好，但较低的层只有一个小的接受域。为了解决这一局限性，我们提出了一种结合大接受域和小接受域的双扩展层。我们进一步将第一阶段的设计与改良阶段分离，以满足这些阶段的不同要求。广泛的评估结果表明，该模型在获取长期相关性和识别动作片段方面是有效的。我们的模型在以下三个数据集上获得了最准确的结果: 50Salads, Georgia Tech Egocentric Activities (GTEA) 和 Breakfast 数据集。

Index Terms—时域动作分割，时域卷积网络

1 介绍

视频中的动作识别是近年来计算机视觉领域的一个研究热点。然而，大部分的努力都集中在对短视频进行分类 [2]–[5]。尽管这些方法在单一活动的视频剪辑上取得了成功，但在包含许多动作片段的长视频上，它们的性能是有限的。由于在监视、机器人技术等许多应用中，对长视频中的活动进行时域分割是至关重要的，因此时域动作分割方法受到了越来越多的关注。时域动作分割方面的早期探索通过将这些模型与滑动窗口 [6]–[8] 相结合来扩展在视频剪辑上取得的成功。这些方法利用不同尺度的时间窗来检测和分类动作片段。然而，这种方法成本高昂，而且不适用于长视频。其他方法在逐帧分类器 [9]–[11] 的基础上使用马尔科夫模型进行粗糙的时域建模。虽然这些方法取得了很好的结果，但它们的速度非常慢，因为它们需要在很长的序列上解决最大化问题。

随着时域卷积网络 (TCNs) 作为一种强大的语音合成时域模型取得成功，许多研究者采用基于 TCN 的模型来解决时域动作分割任务 [12]–[14]。这些模型依靠一个大的接受域，可以更好地捕获视频帧之间的长期相关性。然而，这些模型局限于处理每秒几帧的、非常低的时域分辨率的视频。此外，由于这些方法依赖于时域池化层来扩大接受域，许多用于识别的细粒度信息会丢失。

为了克服以往方法的局限性，我们提出了一个新的模型，也使用时域卷积。与之前的方法相比，所提出的模型能在视频的全时域分辨率上运行，从而获得更好的结果。我们的模型由多个阶段组成，每个阶段输出一个初步预测，然后由下一个阶段改良。我们称之为多阶段时域卷积网络 (MS-TCN)。在每个阶段，我们都会应用一系列扩展的一维卷积，使模型具有参数较少的、大的时域接收域。Figure 1 显示了所提出的多阶段模型的概览。此外，我们在训练中使用平滑损失来惩罚预测中的过度分割错误。

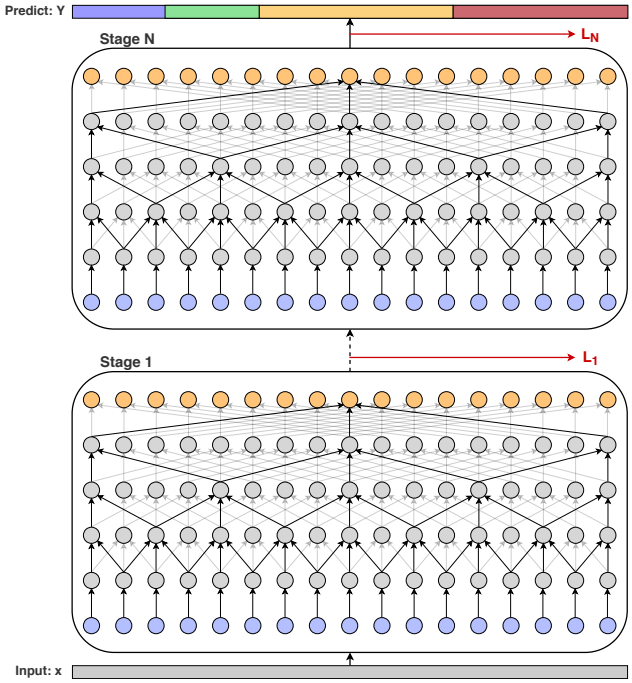


图 1: 多阶段时间卷积网络 (MS-TCN) 概览。每个阶段都会生成一个初始预测，并由下一阶段进行改良。在每个阶段，都会在前一层的激活上应用几个扩展的 1D 卷积。在每个阶段之后添加损失层。

- *S. Li, Y. Abu Farha, and J. Gall are with the University of Bonn, Germany.*
- *Y. Liu and M.-M. Cheng are with the Nankai University, China.*
- *S. Li and Y. Abu Farha contributed equally.*
- *This is a Chinese translated version of paper [1].*
- *E-mails: {lishijie, abufarha, gall}@iai.uni-bonn.de (S. Li, Y. Abu Farha, and J. Gall), vagrantlyun@gmail.com (Y. Liu), cmm@nankai.edu.cn (M.-M. Cheng)*

这项工作引入 MS-TCN 和平滑损失的初步版本已在 [15] 中发表。虽然提出的 MS-TCN 已经取得了良好的性能，但有些设计选择并非最好的。首先，在 MS-TCN 中，较高层的接受域非常大，而较低层的接受域很小。其次，MS-TCN 中第一阶段生成一个初步预测，其余阶段对该预测进行改良。尽管这两个任务之间存在差异，但所有阶段都使用相同的架构。为了解决第一个限制，我们提出了双扩展层 (DDL)，它在每一层结合了大接受域和小接受域。对于第二点，我们将整个体系结构分为两部分: 第一部分是第一阶段，即预测生成阶段，第二部分是预测改良阶段。然后，我们分别制定每个部分的架构，并且不强制所有阶段拥有与 MS-TCN 相同的架构。通过在 MS-TCN 上整合这些设计选择，我们提出了该模型的改进版本，我们将之命名为 MS-TCN++。此外，我们还证明了在不影响准确率的情况下，MS-TCN++ 中改良阶段的参数可以被共享。与 MS-TCN 相比，该模型以少得多的参数实现了更好的性能。我们在 [15] 基础上的贡献分为以下三部分:

- 我们提出了一个结合大接受域和小接受域的双扩张层。
- 我们通过去除预测阶段和改良阶段的耦合来优化 MS-TCN 的架构设计。我们将新模型命名为 MS-TCN++，其效果优于 MS-TCN。
- 我们进一步证明，在 MS-TCN++ 的改良阶段之间共享参数可以在不影响性能的情况下得到更紧凑的模型。

量的评估证明了我们的模型在捕获动作类别之间的长期相关性和产生高质量预测方面的有效性。我们的方法在三个具有挑战性的评判标准上达到了最精准的结果:50Salads [16], Georgia Tech，Egocentric Activities (GTEA) [17]，和 the Breakfast dataset [18]。此外，所提出的模型不受视角限制，在三个数据集上都运行良好，这三个数据集分别由第三人称视图、俯视图和第一人称视图的视频组成。

2 相关工作

时域动作分割已经引起了计算机视觉界的广泛关注。人们提出了许多方法来定位视频中的动作片段或为视频帧指定动作标签。在早期的方法中，主要采用非最大抑制的滑动窗口方法 [6], [7]。然而，这样的方法的计算代价是高昂的，因为模型必须在不同的窗口尺度上进行评估。其他方法基于物体和材料状态的变化 [19] 或基于手和物体之间的交互 [20] 来进行动作建模。Bhattacharya *et al.* [21] 使用视频的时间序列矢量表示形式，使用线性动力系统理论的方法来对复杂动作的时域动力学建模。基于预训练过的、用于重叠时域窗的概念检测器的输出，他们得到这种表示方法。Cheng *et al.* [22] 将视频表示为一个视觉词序列，并采用离散序列的非参数贝叶斯模型对视频序列同时进行分类和分割，对时域相关性进行建模。

尽管之前的方法成功了，但由于它们在捕捉长视频序列的前后关系时失败了，它们的性能是有限的。为了缓解这一问题，许多 proposals 尝试在逐帧分类器上采用高级时域建模。Kuehne *et al.* [9] 使用改进密集轨迹的 Fisher 向量表示视频的帧，然后使用隐马尔可夫模型 (HMM) 对每个动作建模。这些 HMMs 和与前后无关的语法结合在一起进行识别，以确定最可能的动作序列。HMMs 也用于许多其他方法。[23] 将 HMMs 与高斯混合模型 (GMM) 结合作为一种逐帧分类器。然而，由于逐帧分类器无法捕获足够的前后关系来检测动作类，Richard *et al.* [11] 和 Kuehne *et al.* [24] 使用 GRU 而非 [23] 中使用的 GMM。[25] 还使用了隐马尔可夫模型来建模状态之间的转换及其持续时间。Vo 和 Bobick [26] 使用贝叶斯网络来分割活动。它们使用随机的与前后无关的语法和 AND-OR 操作表示动作的组成部分。[27] 提出了一个时域动作检测模型，其包含三个部分: 一个将从视频帧中提取的特征映射到动作概率的动作模型, 一个描述序列级动作概率的语言模型, 最后是一个对不同的动作片段的长度建模的长度模型。为了实现视频分割，他们使用动态规划来寻找三个模型取得最大联合概率的方案。Singh *et al.* [28] 使用双流网络来学习短视频的分块表示。然后将这些表示传递给双向 LSTM，以捕捉不同块之间的相关性。但是，由于采用的是顺序预测，他们的方法速度很慢。在 [29] 中，一个在空间、时间和第一人称视图的流构成的三流架构被引入以学习第一人称视图的特定特性。然后使用多类支持向量机对这些特征进行分类。

多尺度信息对于语音合成 [30] 和图像识别 [31] 很重要。受此启发，研究人员试图将类似的想法用于时域动作分割任务。Lea *et al.* [12] 提出了一种用于动作分割和检测的时域卷积网络。他们的方法遵循编码器-解码器架构，在编码器中进行时域卷积和池化，在解码器中进行上采样和反卷积。虽然使用时域池化令模型能够捕捉长期相关性，但它可能会导致细粒度识别所需的细粒度信息丢失。Lei 和 Todorovic [13] 在 [12] 的基础上改进，使用可变形卷积代替正常卷积，并在编码器-解码器模型中添加了一个残差流。Ding 和 Xu [14] 在编码器-解码器 TCN [12] 的基础上增加了横向连接，并提出了一种预测逐帧动作标签的时域卷积特征金字塔状网络。[12]–[14] 中的所有方法都是对时域分辨率为每秒 1-3 帧的下采样视频进行操作的。与这些方法相反，我们对全时域分辨率进行操作，并使用扩张的卷积来捕捉长期相关性。最近，Mac *et al.* [32] 提出使用可变形卷积和局部一致性约束来学习时空特征。与之相反，在我们的方法中，我们只关注长期的时域建模。

动作检测是一项相关但不同的任务。在这种情况下，目标是在视频大部分没有标记的情况下检测稀疏的动作片段。在这项工作中，我们重点关注对视频密集注释的动作分割。对于动作检测，多种方法遵循两阶段原则。第一阶段是生成 proposals, ，然后在第二阶段对这些 proposals 进行分类和边界调整 [33]–[41]。其他方法将 proposal 的生成和分类结合在一个单阶段架构中，实现一端到一端的训练 [42]–[44]。

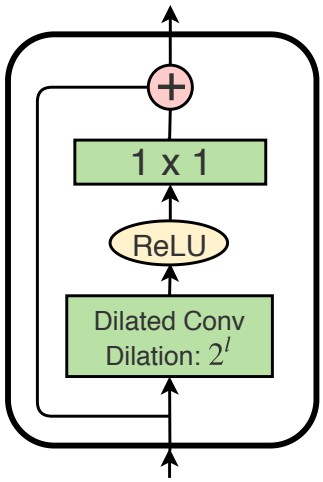


图 2: 扩展残余层概览。在每一层 l，扩展残余层使用具有膨胀因子 2^l 的卷积。

3 时域动作分割

我们引入了一个多阶段时域卷积网络 (MS-TCN) 来完成时域动作分割任务。然后，我们引入了一个新的层，并克服了 MS-TCN 的局限性，提出了一个改进的模型，即 MS-TCN++。给定视频的帧 $x_{1:T} = (x_1, \ldots, x_T)$，我们的目标是推断每个帧的类别标签 $c_{1:T} = (c_1, \ldots, c_T)$，其中 T 为视频长度。首先，我们在 Section 3.1描述了单阶段方法，然后在 Section 3.2讨论多阶段模型。Section 3.3 介绍了双扩展层。在 Section 3.4中，我们分析了 MS-TCN 模型的缺点，并介绍了改进的模型 MS-TCN++。最后，在 Section 3.5中我们描述了所提出的损失函数。

3.1 单阶段时域卷积网络 (SS-TCN)

我们的单阶段模型仅由时域卷积层组成。我们不使用降低时域分辨率的池化层，也不使用强制模型在固定大小的输入上运行并大量增加参数数量的全连接层。我们称这个模型为单阶段时域卷积网络 (SS-TCN). 单阶段 TCN 的第一层是 1×1 卷积层，可调整输入特征的维度以匹配网络中特征图的数量。然后，这层后面是几个扩展一维卷积层。受 wavenet [30] 架构的启发，我们使用在每层加倍的膨胀因子，*i.e.* $1, 2, 4, ..., 512$。所有这些层都有相同数量的卷积滤波器。然而，代替在 wavenet 中使用的因果卷积，我们使用核尺寸为 3 的非因果卷积。每一层对前一层的输出应用一个带有 ReLU 激活的扩展卷积。我们进一步使用剩余连接来促进梯度流动。每一层的操作集合可以如下正规表述：

$$\hat{H}_l = ReLU(W_d * H_{l-1} + b_d), \tag{1}$$

$$H_l = H_{l-1} + W * \hat{H}_l + b, \tag{2}$$

其中，H_l 是第 l 层的输出，$*$ 表示卷积运算，$W_d \in \mathbb{R}^{3 \times D \times D}$ 是核尺寸为 3 的扩展卷积滤波器的权重，D 是卷积滤波器的数量，$W \in \mathbb{R}^{1 \times D \times D}$ 是 1×1 卷积的权重，，$b_d, b \in \mathbb{R}^D$ 是偏差向量。这些操作如图 2所示。使用扩展卷积增加接受域，并不需要通过增加层数或扩大核尺寸来增加参数的数量。由于接受域随着层数成指数增长，我们可以用几层网络获得非常大的接受域，这有助于防止模型过度拟合训练数据。每一层的接受域由以下因素决定

$$ReceptiveField(l) = 2^{l+1} - 1, \tag{3}$$

其中 $l \in [1, L]$ 是层数。注意，这个公式只对大小为 3 的核有效。为了获得输出类别的概率，我们在最后一个扩展卷积层的输出上应用 1×1 卷积，然后是 softmax 激活，*i.e.*

$$Y_t = Softmax(W h_{L,t} + b), \tag{4}$$

其中，Y_t 包含在时间 t 的类别概率，$h_{L,t}$ 是在时间 t 的最后一个扩展卷积层的输出，$W \in \mathbb{R}^{C \times D}$ 和 $b \in \mathbb{R}^C$ 是 1×1 卷积层的权重和偏差。C 是类别个数，D 是卷积滤波器的个数。

3.2 多阶段时域卷积网络 (MS-TCN)

依次使用几个预测器已经在许多任务中展现出显著的性能提升，如人体姿态估计 [45]–[47]。这些堆叠或多阶段架构的思想是按顺序组建几个模型，这样每个模型都直接在前一个模型的输出上运行。这种组合的效果是对前几个阶段预测的逐步改良。受这些架构取得成功的激励，我们引入了一个多阶段时间卷积网络 (MS-TCN) 来完成时域动作分割任务。在这个多阶段模型中，每个阶段从前一个阶段获得一个初始预测，并对其进行改良。第一阶段的输入是视频的逐帧特征，如下所示

$$Y^0 = x_{1:T}, \tag{5}$$

$$Y^s = \mathcal{F}(Y^{s-1}), \tag{6}$$

其中 Y^s 是阶段 s 的输出，$\mathcal{F}$ 是第 3.1节中讨论的单阶段时域卷积网络。用这样的多级架构有助于提供更多的前后信息来预测每一帧的类别标签。此外，由于每个阶段的输出是一个初始预测，网络能够捕捉动作类别之间的依赖关系并学习合理的动作序列，这有助于减少过度分割的错误。

注意，下一阶段的输入只是视频帧间的概率，没有任何附加特征。我们将在实验中展示向下一阶段的输入中添加特征将对预测的质量造成怎样的影响。

3.3 双扩展层 (DDL)

在 MS-TCN 的扩展卷积层中，膨胀因子随着层数的增加而增加。虽然这导致较高层的接受域很大，而较低层的接受域仍然很小。此外，由于较大的膨胀因子，MS-TCN 中的较高层在非常长的时间步长上应用卷积。为了克服这一问题，我们提出了双扩展层 (DDL)。DDL 结合了两个具有不同膨胀因子的卷积，而不是只有一个膨胀卷积。第一个卷积在较低的层中具有较低的膨胀因子，并且随着层数的增加而指数增加。而

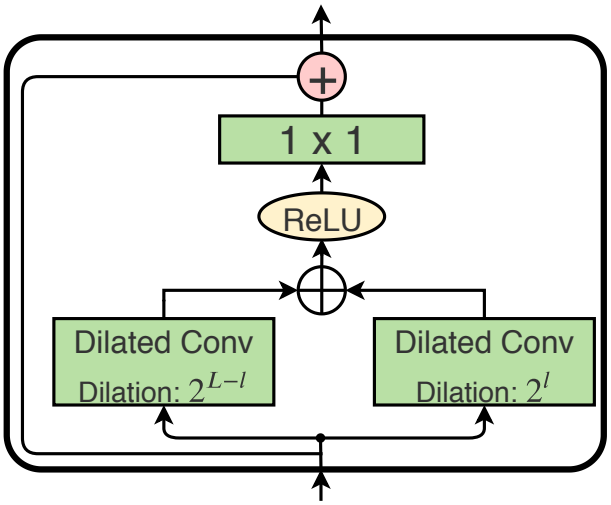


图 3: 双扩展层 (DDL) 概览。在每一层 l，DDL 分别使用两个具有膨胀因子 2^l 和 2^{L-l} 的卷积，其中 L 是网络中的层数。

对于第二个卷积，我们从较低层的大膨胀因子开始，并随着层数的增加而指数减小。每一层的操作集合可以正式描述如下：

$$\hat{H}_{l,d_1} = W_{d_1} * H_{l-1} + b_{d_1}, \tag{7}$$

$$\hat{H}_{l,d_2} = W_{d_2} * H_{l-1} + b_{d_2}, \tag{8}$$

$$\hat{H}_l = ReLU([\hat{H}_{l,d_1}, \hat{H}_{l,d_2}]), \tag{9}$$

$$H_l = H_{l-1} + W * \hat{H}_l + b, \tag{10}$$

其中 $W_{d_1}, W_{d_2} \in \mathbb{R}^{3\times D\times D}$ 分别是具有膨胀因子 2^l 和 2^{L-l} 的扩展卷积的权重，$W \in \mathbb{R}^{1\times 2D\times D}$ 是 1×1 卷积的权重，$b_{d_1}, b_{d_2}, b \in \mathbb{R}^D$ 是偏差向量。在 (9) 中，$\hat{H}_{l,d_1}$ 和 $\hat{H}_{l,d_2}$ 是相连的。图 3显示了双扩展层的概况。

虽然双扩展层结合了来自输入序列的局部和全局信息，但是在文献中还有其他技术用于融合多尺度特征，例如特征金字塔网络 (FPN) [48]。虽然将 FPN 应用于时域动作分割已经取得了成功 [14]，但这些方法只有非常有限的接受域。此外，FPN 的多尺度特征是通过应用池化操作获得的，这些池化操作会导致时域分割所必需的细粒度信息丢失。相反，DDL 结合了多尺度特征，同时保持了输入序列的时域分辨率。

3.4 MS-TCN++

在这一节中，我们介绍 MS-TCN++，它利用提出的双扩展层来改善 MS-TCN。类似于 MS-TCN，MS-TCN++ 中的第一阶段负责生成初始预测，而其余阶段逐步改良该预测。对于预测生成阶段，我们采用了具有双扩张层的 SS-TCN(图 3)，双扩展层取代了在 SS-TCN 中原本使用的简单扩展残余层 (图 2)。使用 DDL 使预测生成阶段能够捕捉所有层中的局部和全局特征，从而实现更好的预测。由于改良比预测生成更容易，我们使用具有扩展残余层的 SS-TCN 架构实现改良阶段。在我们的实验中，我们证明只在第一阶段使用 DDL 效果最好。图 4显示了提出的 MS-TCN++ 的概览。

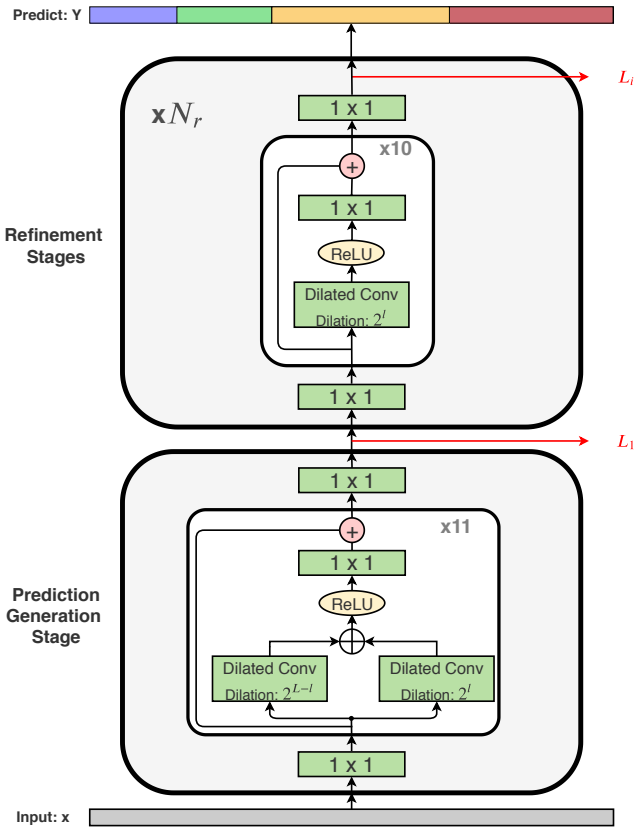


图 4: MS-TCN++ 概览。第一阶段采用具有双扩展层的 SS-TCN 模型。这个阶段生成一个初始预测，该预测通过一组 N_r 个改良阶段逐步改良。对于改良阶段，使用具有扩展残余层的 SS-TCN。在每个阶段之后添加损失层。

虽然增加更多的阶段会逐步改良预测，但也会大幅增加参数的数量。但是，由于改良阶段具有相同的作用，因此可以共享它们的参数以获得更紧凑的模型。在实验中，我们表明在改良阶段之间共享参数会显著减少参数的数量，而准确性只会略有下降。

3.5 损失函数

对于损失函数，我们使用分类损失和平滑损失的组合。对于分类损失，我们使用交叉熵损失

$$\mathcal{L}_{cls} = \frac{1}{T}\sum_t -log(y_{t,c}), \tag{11}$$

其中 $y_{t,c}$ 是在时间 t 的真实标签 c 的预测概率。

虽然交叉熵损失已经表现良好，但是我们发现一些视频的预测包含一些过度分割错误。为了进一步提高预测的质量，我们使用了额外的平滑损失来减少这种过度分割的错误。对于这种损失，我们在帧间对数概率上使用截断均方误差

$$\mathcal{L}_{T-MSE} = \frac{1}{TC}\sum_{t,c}\tilde{\Delta}^2_{t,c}, \tag{12}$$

$$\tilde{\Delta}_{t,c} = \begin{cases} \Delta_{t,c} & : \Delta_{t,c} \leq \tau \\ \tau & : otherwise \end{cases}, \tag{13}$$

$$\Delta_{t,c} = |log\ y_{t,c} - log\ y_{t-1,c}|, \tag{14}$$

其中，T 是视频长度，C 是类别数，$y_{t,c}$ 是在时间 t 时类别 c 的预测概率。

注意，梯度仅针对 $y_{t,c}$ 计算，而 $y_{t-1,c}$ 不被视为模型参数的函数。这种损失类似于 KullbackLeibler(KL) 散度损失，其中

$$\mathcal{L}_{KL} = \frac{1}{T}\sum_{t,c} y_{t-1,c}(log\ y_{t-1,c} - log\ y_{t,c}). \tag{15}$$

然而，我们发现截断均方误差 ($\mathcal{L}_{T-MSE}$) (12) 更能减少过度分割误差。我们将在实验中比较 KL 损失和我们所提出的损失。

单阶段的最终损失函数是上述损失的组合

$$\mathcal{L}_s = \mathcal{L}_{cls} + \lambda\mathcal{L}_{T-MSE}, \tag{16}$$

其中 λ 是确定不同损失贡献的模型超参数。最后，为了训练完整的模型，我们最小化所有阶段的损失总和

$$\mathcal{L} = \sum_s \mathcal{L}_s. \tag{17}$$

3.6 实现细节

对于 MS-TCN 和 MS-TCN++ 来说，我们使用了一个包含四个阶段的多阶段架构。所有阶段对于 MS-TCN 是相同的，而 MS-TCN++ 中的阶段包括一个预测生成阶段和三个改良阶段。MS-TCN 中的每个阶段和 MS-TCN++ 中的改良阶段包含十个扩展卷积层。对于 MS-TCN++ 中的预测生成阶段，我们使用了 11 层。每层后使用概率为 0.5 的 dropout。我们在模型的所有层中将过滤器的数量设置为 64，过滤器的大小为 3。对于损失函数，我们设置 $\tau = 4$，$\lambda = 0.15$。在所有实验中，我们使用学习率为 0.0005 的 Adam 优化器。

4 实验

数据集。我们在三个具有挑战性的数据集上评估了所提出的模型:50Salads [16]，Georgia Tech Egocentric Activities (GTEA) [17]，和 the Breakfast dataset [18]。表 1 显示了这些数据集的摘要。

50Salads 数据集包含 17 个动作类的 50 个视频。视频是从俯视视角录制的。平均每个视频包含 20 个动作实例，时长 6.4 分钟。如数据集名称所示，视频描述了沙拉的准备活动。这些活动由 25 名演员表演，每个演员准备两个不同的沙拉。对于评估，我们使用五重交叉验证，并报告平均值，如 [16] 所示。

GTEA 数据集包含 28 个视频，对应于 7 种不同的活动，如准备咖啡或奶酪三明治，这些视频由 4 位演员完成。这个数据集包含由安装在演员头上的摄像机记录的第一人称视图的视频。视频的帧已被标注，注释包含 11 个动作类别（包括背景）。平均每个视频有 20 个动作实例。我们通过移出一个实例来使用交叉验证进行评估。

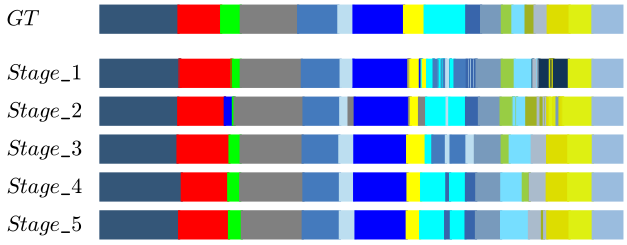


图 5: 来自 50Salads 数据集的定性结果，用于比较不同的阶段数。

The **Breakfast**dataset 是三个数据集中最大的，有着 1,712 个第三人称视图的视频。这些视频在 18 个不同的厨房中录制，展示了与早餐准备相关的活动。总的来说，共有 48 个不同的动作，每个视频平均包含 6 个动作实例。为了评估，我们使用 [18] 中提出的标准 4 分割，并报告平均值。

对于所有数据集，我们提取视频帧的 I3D [4] 特征，并将这些特征用作我们模型的输入。对于 GTEA 和 Breakfast 数据集，我们使用 15 fps 的时域视频分辨率，而对于 50Salads，我们将特征从 30 fps 下采样到到 15 fps，来与其他数据集一致。

评估指标。为了评估，我们报告了帧级的准确性 (Acc)、片段编辑距离和分段 F1 在重叠阈值 10%, 25% 和 50% 时的分数，用 $F1@\{10, 25, 50\}$ 表示。重叠阈值是基于交并比 (IoU) 确定的。虽然帧级准确性是动作分割中最常用的度量标准，但是长动作类对该度量的影响比短动作类大，并且过度分割错误的影响非常小。因此，我们使用分段 F1 分数作为预测质量的衡量标准，在 [12] 提出的。

4.1 阶段数目的影响

我们通过展示使用多阶段架构 (MS-TCN) 的效果来开始我们的评估。表 2 显示了单阶段模型与具有不同阶段数目的多阶段模型的比较结果。如表中所示，所有这些模型都达到了同等的的帧级准确率。然而，预测的质量大为不同。观察这些模型的分段编辑距离和 F1 得分，我们可以看出单阶段模型产生了大量的过度分割错误，因为它的 F1 得分较低。另一方面，使用多级架构可以减少这些错误并提高 F1 得分。当我们使用两个或三个阶段时，其效果是显著的，这极大地提高了准确性。增加第四个阶段仍然会改良结果，但效果不如前几个阶段显著。然而，通过增加第五阶段，我们可以看到性能开始下降。这可能是参数数量增加导致的过拟合问题。多阶段架构的效果也可以在图 5所示的定性结果中看到。增加更多的阶段会使预测的逐步改良。在其余的实验中，我们使用了具有四个阶段的多阶段 TCN。

	# videos	# action classes	view	description
50Salads	50	17	top-view	salad preparation activities
GTEA	28	11	egocentric	7 different activities, like preparing coffee or cheese sandwich
Breakfast	1712	48	third person view	breakfast preparation related activities in 18 different kitchens

表 1: 实验中所使用的数据集的总结。

	F1@{10,25,50}			Edit	Acc
SS-TCN	27.0	25.3	21.5	20.5	78.2
MS-TCN (2 stages)	55.5	52.9	47.3	47.9	79.8
MS-TCN (3 stages)	71.5	68.6	61.1	64.0	78.6
MS-TCN (4 stages)	76.3	**74.0**	**64.5**	67.9	**80.7**
MS-TCN (5 stages)	**76.4**	73.4	63.6	**69.2**	79.5

表 2: 阶段数对模型在 50Salads 数据集上表现的影响。

	F1@{10,25,50}			Edit	Acc
SS-TCN (48 layers)	49.0	46.4	40.2	40.7	78.0
MS-TCN	**76.3**	**74.0**	**64.5**	**67.9**	**80.7**

表 3: 在 50Salads 数据集上比较 MS-TCN 和深度 SS-TCN。

4.2 多阶段时域卷积网络 vs. 更深的单阶段时域卷积网络

在前一节中，我们已经看到我们的多阶段架构比单阶段架构更好。然而，这种比较并没有显示出这种改进是因为多阶段架构还是因为增加更多阶段时参数数量的增加。为了公平比较，我们训练一个单阶段模型，该模型具有与多阶段模型同样数量的参数。我们的 MS-TCN 中的每一阶段包含 12 层 (10 个扩展卷积层、一个 1×1 卷积层和一个 softmax 层)，我们训练了一个具有 48 层的 SS-TCN，这与具有四个阶段的 MS-TCN 的层数相同。对于扩展卷积，在我们的 MS-TCN 中，我们使用了相似的膨胀因子。也就是说，我们从 1 的膨胀系数开始，在每一层增加一倍，直至系数达到 512，然后我们再次从 1 开始。如表 3所示，我们的多阶段架构的性能优于单阶段架构副本，性能提升高达 27%。这突出了多阶段架构在提高预测质量方面的影响。

4.3 比较不同的损失函数

对于损失函数，我们使用交叉熵损失和帧级概率对数的截断均方损失的组合来确保平滑预测，交叉熵损失是分类任务的常见做法。与单独使用交叉熵损失相比，平滑损失略微提高了帧级准确性，同时我们发现这种损失产生的过度分割错误要少得多。表 4和图 6显示了这些损失的比较。如表 4所示，提出的损失函数实现了更好的 F1 和编辑分数，提高绝对高达 5%。这表明，与交叉熵相比，我们的损失函数产生的过度分割错误更少，因为它迫使连续帧具有相似的类别概率，从而有更平滑的输出。

	F1@{10,25,50}			Edit	Acc
$\mathcal{L}_{cls}$	71.3	69.7	60.7	64.2	79.9
$\mathcal{L}_{cls}+\lambda\mathcal{L}_{KL}$	71.9	69.3	60.1	64.6	80.2
$\mathcal{L}_{cls}+\lambda\mathcal{L}_{T-MSE}$	**76.3**	**74.0**	**64.5**	**67.9**	**80.7**

表 4: 在 50Salads 数据集上比较不同的损失函数。

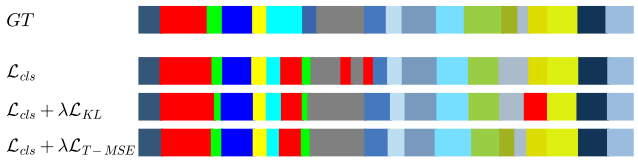


图 6: 50Salads 数据集的定性结果，用于比较不同的损失函数。

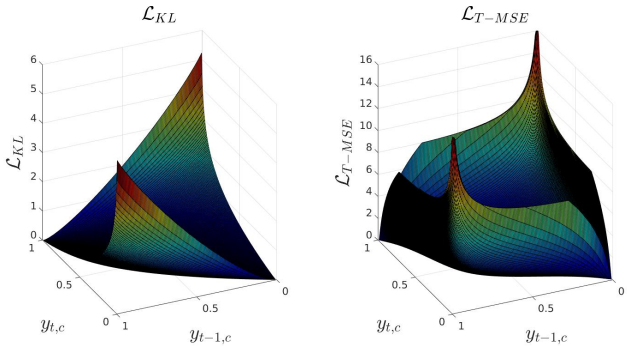


图 7: 库勒贝克-莱布勒 (KL) 散度损失 ($\mathcal{L}_{KL}$) 和两类情况下的截断均方损失 ($\mathcal{L}_{T-MSE}$) 的损失面。$y_{t,c}$ 是 c 类预测概率，$y_{t-1,c}$ 是对应于该类的目标概率。

惩罚概率对数的差异类似于库勒贝克-莱布勒 (KL) 散度损失，它测量两个概率分布之间的差异。然而，结果表明，如表 4 和图 6所示，我们提出的损失函数比使用 KL 损失有更好的结果。这背后的原因是，KL 散度损失对目标概率和预测概率之间的差异非常小的情况不进行惩罚。然而，我们提出的损失函数也对小的差异进行惩罚。注意，与 KL 损失相反，我们提出的损失是对称的。图 7 显示了两类情况下 KL 损失和我们提出的截断均方损失的曲面图。我们也尝试了对称版本的 KL 损失，但它的表现比 KL 损失更差。

Impact of λ	F1@{10,25,50}			Edit	Acc
MS-TCN ($\lambda = 0.05$, $\tau = 4$)	74.1	71.7	62.4	66.6	80.0
MS-TCN ($\lambda = 0.15$, $\tau = 4$)	**76.3**	**74.0**	**64.5**	67.9	**80.7**
MS-TCN ($\lambda = 0.25$, $\tau = 4$)	74.7	72.4	63.7	**68.1**	78.9

Impact of τ	F1@{10,25,50}			Edit	Acc
MS-TCN ($\lambda = 0.15$, $\tau = 3$)	74.2	72.1	62.2	67.1	79.4
MS-TCN ($\lambda = 0.15$, $\tau = 4$)	**76.3**	**74.0**	**64.5**	**67.9**	**80.7**
MS-TCN ($\lambda = 0.15$, $\tau = 5$)	66.6	63.7	54.7	60.0	74.0

表 5: λ 和 τ 对模型在 50Salads 数据集上表现的影响。

	F1@{10,25,50}			Edit	Acc
Probabilities and features	56.2	53.7	45.8	47.6	76.8
Probabilities only	**76.3**	**74.0**	**64.5**	**67.9**	**80.7**

表 6: 将特征传递到 50Salads 数据集中更高阶段的效果。

4.4 λ 和 τ 的影响

所提出的平滑损耗的效果由两个超参数控制:λ 和 τ。在本节中，我们将研究这些参数的影响，并了解它们如何影响该模型的性能。

λ 的影响: 在所有实验中，我们设置 $\lambda = 0.15$。为了分析这个参数的影响，我们用不同的 λ 值训练不同的模型。如表 5所示，λ 对性能的影响非常小。将 λ 减小到 0.05 仍可提高性能，但不如默认设置 ($\lambda = 0.15$) 表现好。将其值增加到 $\lambda = 0.25$ 也会导致性能下降。这种性能下降是由于平滑损失项严重惩罚了帧间标签的变化，从而影响了动作片段之间检测到的边界。

τ 的影响: 这个超参数定义了截断平滑损失的阈值。我们的默认值是 $\tau = 4$。将该值降低到 $\tau = 3$ 仍然比交叉熵基线有所进步，而设置 $\tau = 5$ 会导致性能大幅下降。这主要是因为当 τ 太高时，平滑损失项惩罚了模型非常确信连续帧属于两个不同类别的情况，这确实降低了模型检测动作片段之间的真实边界的能力。

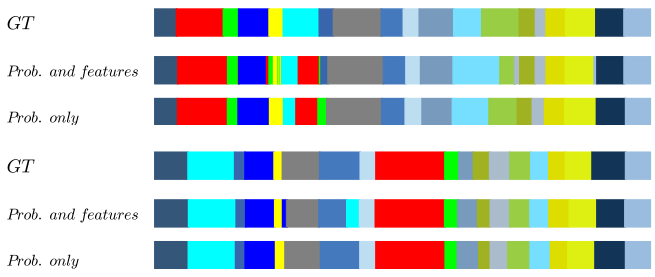


图 8: 来自 50 沙拉数据集的两个视频的定性结果显示了将特征传递到更高阶段的效果。

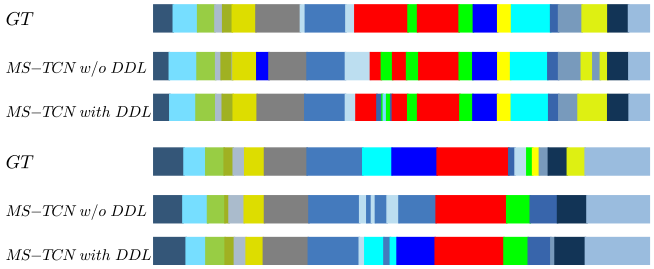


图 9: 来自 50Salads 数据集的两个视频的定性结果显示了双扩展层的影响。

4.5 传递特征到更高阶段的效果

在所提出的多阶段 TCN 中，对更高阶段的输入仅仅是逐帧概率。然而，在用于人体姿态估计的多级架构中，附加特征通常与前一阶段的输出热图相关。因此，在本实验中，我们分析了组合附加特征到更高阶段概率输入的影响。为此，我们训练了两个多阶段卷积神经网络: 一个只使用预测的帧级概率作为下一阶段的输入，而对于第二个模型，我们将每一阶段中最后一个扩展卷积层的输出连接到下一阶段的输入概率。如表 6所示，将特征连接到输入概率会导致 F1 分数和片段编辑距离大幅下降 (约 20%)。我们认为性能下降的原因是许多动作类别具有相似的外观和动作。通过在每个阶段添加这样的类别特征，模型会发生混乱并且产生对应于过度分割效果的小的单独的错误检测动作片段。仅传递概率迫使模型关注由概率明确表示的相邻标签的信息。这种影响也可以在图 8所示的定性结果中看到。

4.6 MS-TCN++ vs. MS-TCN

在本节中，我们比较了两种多级架构:MS-TCN++ 和 MS-TCN。与 MS-TCN 相比，MS-TCN++ 在第一阶段使用双扩展层 (DDL)。表 7 显示了 50Salad 数据集上两种架构的运行结果。如表所示，MS-TCN++ 的表现优于 MS-TCN，性能提升高达 6.4%。这强调了在预测生成阶段通过利用 MS-TCN++ 中的 DDL 来组合局部和全局表示的重要性。为了研究在所有阶段使用 DDL 的影响，我们还训练了一个在所有阶段都使用 DDL 的 MS-TCN。如表 7所示，MS-TCN++ 在各个阶段的表现都优于使用 DDL 的 MS-TCN。这表明改良阶段的设计和预测生成阶段的去耦合是至关重要的。通过 DDL 使用全局信息对预测生成阶段至关重要，而改良阶段更多地关注局部信息。通过向改良阶段添加 DDL，准确率甚至会因过度拟合而下降。注意，在所有阶段使用 DDL 的性能要优于 MS-TCN，高达 2.8%。这进一步凸显了 DDL 的优势。DDL 的影响在图 9所示的定性结果中也很显著。

4.7 层数的影响

对于 MS-TCN 和 MS-TCN++ 中的改良阶段，我们将每个阶段的层数固定为 10 层。而对于 MS-TCN++ 中的预测生成阶

	F1@{10,25,50}			Edit	Acc
MS-TCN	76.3	74.0	64.5	67.9	80.7
MS-TCN with DDL	77.3	75.0	67.3	69.8	82.4
MS-TCN++	**80.7**	**78.5**	**70.1**	**74.3**	**83.7**

表 7: 在 50Salads 数据集上，MS-TCN++ vs. MS-TCN vs. 使用 DDL 的 MS-TCN。

	F1@{10,25,50}			Edit	Acc
$L = 6$	53.2	48.3	39.0	46.2	63.7
$L = 8$	66.4	63.7	52.8	60.1	73.9
$L = 10$	76.3	74.0	64.5	67.9	**80.7**
$L = 11$	76.7	74.2	65.5	**69.7**	80.4
$L = 12$	**77.8**	**75.2**	**66.9**	69.6	80.5

表 8: MS-TCN 每一阶段的层数 (L) 对 50Salads 数据集的影响。

段，我们将层数设置为 11。在本节中，我们研究这些参数的影响。表 8显示了 MS-TCN 中阶段的层数 (L) 对其在 50Salad 数据集上表现的影响。将 L 从 8 增加到 10 可以显著提高性能。这主要是因为接受域的增加。层数超过 10 层 ($L = 11$, $L = 12$) 不会提高帧级精度，但会略微提高 F1 得分。我们还试图只在 MS-TCN 的改良阶段改变层数。如表 9所示，这没有显著影响，使用 10 层可以获得最佳性能。对于 MS-TCN++ 中的改良阶段，层数 L_r 对性能也没有显著影响。为了与 [15] 保持一致，我们将 L_r 设置为 $L_r = 10$，这在表 9中所有评估指标上实现了合理的性能权衡。可以在表 10中发现类似的行为，在预测生成阶段的层数 L_g 取得 $L_g = 11$ 时达到最佳性能。一般来说，与 MS-TCN++ 相比，每个阶段的层数对 MS-TCN 的影响更大。这两种模型之间的主要区别是在 MS-TCN++ 中使用的双扩展层，这表明双重膨胀层可以更好地捕捉局部和全局特征，以生成更好的预测。

		F1@{10,25,50}			Edit	Acc
MS-TCN	$L_r = 6$	74.3	71.5	62.8	66.0	78.6
	$L_r = 8$	75.4	72.4	64.3	**68.0**	79.5
	$L_r = 10$	**76.3**	**74.0**	**64.5**	67.9	**80.7**
	$L_r = 11$	75.0	72.0	63.5	67.6	80.3
	$L_r = 12$	74.1	71.2	62.3	65.7	79.1
MS-TCN++	$L_r = 6$	78.2	75.6	67.7	69.6	82.3
	$L_r = 8$	**80.9**	78.2	**70.2**	73.4	82.9
	$L_r = 10$	80.7	**78.5**	70.1	**74.3**	**83.7**
	$L_r = 11$	80.5	78.3	70.0	72.6	83.4
	$L_r = 12$	79.4	76.9	69.2	71.3	83.5

表 9: 每个改良阶段的层数 (L_r) 对在 50Salads 数据集上表现的影响。

	F1@{10,25,50}			Edit	Acc
$L_g = 6$	74.3	71.6	63.5	67.8	78.5
$L_g = 8$	77.4	75.3	67.8	70.3	80.8
$L_g = 10$	79.8	77.9	**71.0**	72.5	83.1
$L_g = 11$	**80.7**	**78.5**	70.1	**74.3**	**83.7**
$L_g = 12$	78.9	76.6	67.6	70.8	83.2

表 10: MS-TCN++ 预测生成阶段的层数 (L_g) 对 50Salads 数据集的影响。

	Duration	F1@{10,25,50}			Edit	Acc
MS-TCN	< 1 min	89.6	87.9	77.0	82.5	76.6
	$1 - 1.5$ min	85.9	84.3	71.9	80.7	76.4
	≥ 1.5 min	81.2	76.5	58.4	71.8	75.9
MS-TCN++	< 1 min	90.4	90.4	80.8	84.4	79.3
	$1 - 1.5$ min	88.7	85.8	75.1	83.6	79.3
	≥ 1.5 min	80.8	78.8	63.3	76.1	77.2

表 11: 改良阶段数对模型在 50Salads 数据集上表现的影响。

4.8 大的接受域对短视频的影响

为了研究大的接受域对短视频的影响，我们根据它们的持续时间在三组视频上进行了 MS-TCN 和 MS-TCN++ 的评估。对于此评估，我们使用 GTEA 数据集，因为与其他数据集相比，它包含更短的视频。如表 11所示，MS-TCN 和 MS-TCN++ 在长短视频上都表现良好。然而，由于接受域有限，在较长的视频上的表现稍差。MS-TCN++ 相对于 MS-TCN 的提升在短视频和长视频上都很显著。

4.9 改良阶段数的影响

我们将 MS-TCN ++ 中的改良阶段数 N_r 设置为 3，从而得出一个总共有 4 个阶段的模型。表 12 显示了改良阶段数对 50Salads 数据集的影响。仅使用预测生成阶段 ($N_r = 0$) 会导致模型只有相对较低的性能，但它比单阶段 TCN 好得多 (表 2)。添加更多的改良阶段可逐步提高性能。但是，添加 3 个以上的改良阶段并不能提供额外的提升。

4.10 参数共享的影响

MS-TCN++ 由 1 个预测生成阶段和 3 个改良阶段组成。虽然增加更多的阶段会带来更好的性能，但也会增加参数的数

	N_r	F1@{10,25,50}			Edit	Acc
MS-TCN++	0	51.0	48.4	40.7	40.4	80.7
MS-TCN++	1	70.7	68.2	59.7	62.0	82.4
MS-TCN++	2	77.8	75.1	66.9	69.4	82.5
MS-TCN++	3	**80.7**	78.5	**70.1**	**74.3**	**83.7**
MS-TCN++	4	80.6	**78.7**	**70.1**	73.1	82.4

表 12: 改良阶段共享参数对 50Salads 数据集的影响。

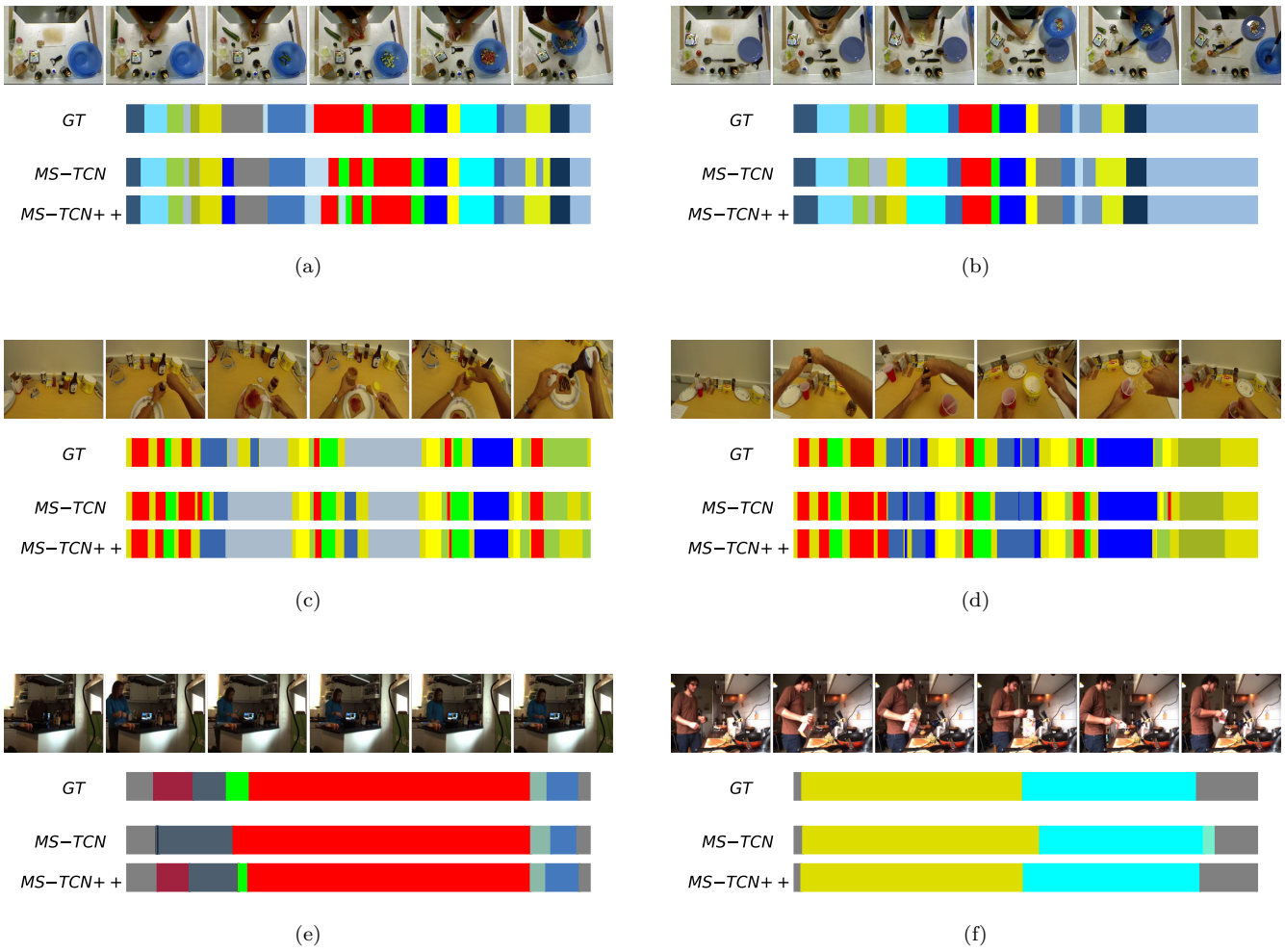


图 10: 在 (a)(b)50Salads，(c)(d) GTEA 和 (e)(f)the Breakfast 数据集上的时域动作分割任务的定性结果。

	F1@{10,25,50}			Edit	Acc	# param.(m)
MS-TCN	76.3	74.0	64.5	67.9	80.7	0.80
MS-TCN++	**80.7**	**78.5**	**70.1**	**74.3**	**83.7**	0.99
MS-TCN++(sh)	78.7	76.6	68.3	70.7	82.2	0.66

表 13: 时域分辨率对 50Salads 数据集的影响。

量。改良阶段原则上共享相同的任务，因此它们可以共享参数是很直观的。表 13显示了改良阶段之间共享参数的影响。共享参数显著减少了参数数量，而性能仅略有下降。对于具有 3 个改良阶段的 MS-TCN++ 来说，共享参数将参数总数减少到原始模型中总参数的 66% 左右。如表所示，尽管参数较少，但共享参数的 MS-TCN++ 的性能优于 MS-TCN，可达 3.8%。

4.11 时域分辨率的影响

过去的时域模型以每秒 1-3 帧的低时域分辨率运行 [12]–[14]。与之相反，我们的方法能够处理 15 fps 的更高的分辨率。在这个实验中，我们对 1fps 的低时间分辨率下有和无参数共享的 MS-TCN 和 MS-TCN++ 进行了评估。如表 14所示，两种模型都能够处理低时域分辨率和高时域分辨率。虽然降低 MS-TCN 的时域分辨率会产生更好的编辑距离和分段 F1 得分，但使用更高的分辨率会产生更好的帧级准确率。在较低的时域分辨率下运行使得 MS-TCN 更不容易出现过度分割问题，这反映在更好的编辑和 F1 得分上。由于双扩展层，MS-TCN++ 从更高的时域分辨率中获益更多，并且降低 MS-TCN++ 的时域分辨率对所有评估指标的影响是显著的。注意，即使在 MS-TCN++ 中共享改良阶段的参数的情况下，使用更高的时域分辨率也会产生更好的性能。

4.12 微调特征的影响

在我们的实验中，我们使用 I3D 特征而不进行微调。表 15显示了进行微调对模型在 GTEA 数据集上表现的影响。我们的两个多级架构 MS-TCN 和 MS-TCN++，无论有没有微调，都显著优于单级架构。当 MS-TCN++ 中的改良阶段的参数共享时，这也同样成立。微调改善了结果，但是对动作分割微调

	F1@{10,25,50}			Edit	Acc
MS-TCN (1 fps)	**77.8**	**74.9**	64.0	**70.7**	78.6
MS-TCN (15 fps)	76.3	74.0	**64.5**	67.9	**80.7**
MS-TCN++ (1 fps)	80.4	**78.7**	68.6	73.3	81.1
MS-TCN++ (15 fps)	**80.7**	78.5	**70.1**	**74.3**	**83.7**
MS-TCN++(sh) (1 fps)	77.0	73.8	64.0	69.1	80.8
MS-TCN++(sh) (15 fps)	**78.7**	**76.6**	**68.3**	**70.7**	**82.2**

表 14: Impact of temporal resolution on the 50Salads dataset.

的效果低于对动作识别微调的效果。这是可以预期的，因为时域模型对于分割比对识别而言更为重要。

注意，如果不进行微调，则共享参数可以在 GTEA 上获得更好的结果。这主要是由于减少了参数数量，这防止了模型过度拟合训练数据，尤其是对于 GTEA 这样的小型数据集。

		F1@{10,25,50}			Edit	Acc
w/o FT	SS-TCN	62.8	60.0	48.1	55.0	73.3
	MS-TCN	85.8	83.4	69.8	79.0	76.3
	MS-TCN++	87.0	85.2	73.5	82.0	78.7
	MS-TCN++(sh)	**87.8**	**86.2**	**74.4**	**82.6**	**78.9**
with FT	SS-TCN	69.5	64.9	55.8	61.1	75.3
	MS-TCN	87.5	85.4	74.6	81.4	79.2
	MS-TCN++	**88.8**	85.7	**76.0**	**83.5**	**80.1**
	MS-TCN++(sh)	88.2	**86.2**	75.9	83.0	79.7

表 15: 微调对 GTEA 数据集的影响。

4.13 与最先进方法的比较

在这一节中，我们将以下三个数据集上，将提出的模型与最先进的方法进行比较：50Salads, Georgia Tech Egocentric Activities (GTEA), 和 the Breakfast datasets。结果如表 16所示。如表中所示，我们的模型在三个数据集和三个评估指标（F1 分数、片段编辑距离和帧级准确率 (Acc)）上均优于目前最先进的方法，且有很大的性能提升 (在 50Salads 数据集上帧级准确率的提升高达 11.6%)。各模型在三个数据集的定性结果如图 10所示。注意，所有报告的结果都是使用 I3D 特征获得的。为了分析使用不同类型特征的效果，我们使用改进的密集轨迹 (IDT) 特征在 the Breakfast dataset 上对 MS-TCN 进行评估，IDT 是 the Breakfast dataset 的常用特征。如表 16所示，这些特性的影响非常小。使用 I3D 特征时，逐帧准确率和编辑距离稍好一些，但与 I3D 相比，使用 IDT 特征时，该模型可获得更好的 F1 分数。这主要是因为 I3D 特征同时对运动和外观编码，而 IDT 特征只对运动编码。对于 Breakfast 这样的数据集，使用外观信息对性能没有帮助，因为外观没法给出判断所进行动作的有力证据。这可以从图 10所示的定性结果中看出。视频帧间具有非常相似的外观。因此，额外的外观特征对识别活动没有帮助。如表 16所示，共享改良阶段

50Salads	F1@{10,25,50}			Edit	Acc
Spatial CNN [10]	32.3	27.1	18.9	24.8	54.9
IDT+LM [27]	44.4	38.9	27.8	45.8	48.7
Bi-LSTM [28]	62.6	58.3	47.0	55.6	55.7
Dilated TCN [12]	52.2	47.6	37.4	43.1	59.3
ST-CNN [10]	55.9	49.6	37.1	45.9	59.4
TUnet [49]	59.3	55.6	44.8	50.6	60.6
ED-TCN [12]	68.0	63.9	52.6	52.6	64.7
TResNet [50]	69.2	65.0	54.4	60.5	66.0
TRN [13]	70.2	65.4	56.3	63.7	66.9
TDRN+UNet [13]	69.6	65.0	53.6	62.2	66.1
TDRN [13]	72.9	68.5	57.2	66.0	68.1
LCDC+ED-TCN [32]	73.8	-	-	66.9	72.1
MS-TCN [15]	76.3	74.0	64.5	67.9	80.7
MS-TCN++(sh)	78.7	76.6	68.3	70.7	82.2
MS-TCN++	**80.7**	**78.5**	**70.1**	**74.3**	**83.7**

GTEA	F1@{10,25,50}			Edit	Acc
Spatial CNN [10]	41.8	36.0	25.1	-	54.1
Bi-LSTM [28]	66.5	59.0	43.6	-	55.5
Dilated TCN [12]	58.8	52.2	42.2	-	58.3
ST-CNN [10]	58.7	54.4	41.9	-	60.6
TUnet [49]	67.1	63.7	51.9	60.3	59.9
ED-TCN [12]	72.2	69.3	56.0	-	64.0
LCDC+ED-TCN [32]	75.4	-	-	72.8	65.3
TResNet [50]	74.1	69.9	57.6	64.4	65.8
TRN [13]	77.4	71.3	59.1	72.2	67.8
TDRN+UNet [13]	78.1	73.8	62.2	73.7	69.3
TDRN [13]	79.2	74.4	62.7	74.1	70.1
MS-TCN [15]	87.5	85.4	74.6	81.4	79.2
MS-TCN++(sh)	88.2	**86.2**	75.9	83.0	79.7
MS-TCN++	**88.8**	85.7	**76.0**	**83.5**	**80.1**

Breakfast	F1@{10,25,50}			Edit	Acc
ED-TCN [12]*	-	-	-	-	43.3
HTK [23]	-	-	-	-	50.7
TCFPN [14]	-	-	-	-	52.0
HTK(64) [9]	-	-	-	-	56.3
GRU [11]*	-	-	-	-	60.6
GRU+length prior [24]	-	-	-	-	61.3
MS-TCN (IDT) [15]	58.2	52.9	40.8	61.4	65.1
MS-TCN (I3D) [15]	52.6	48.1	37.9	61.7	66.3
MS-TCN++(I3D) (sh)	63.3	57.7	44.5	64.9	67.3
MS-TCN++ (I3D)	**64.1**	**58.6**	**45.9**	**65.6**	**67.6**

表 16: 与最先进的模型在 50Salads、GTEA 和 the Breakfast 数据集上进行比较。(* 来自 [14])

的参数实现了与 MS-TCN++ 相似的性能，但是它需要的参数减少了大约 66% ，如表 13所示。

由于我们的方法不使用任何循环层，它们在训练和测试期间都非常快。在单个 GTX 1080Ti GPU 上，在 50Salads 数据集上训练 50 代的 MS-TCN++ 只需要 10 0 分钟，而训练具有 64 维隐藏状态的 Bi-LSTM 单个单元需要 35 分钟。这

是由于 LSTM 的顺序预测，其中在任何时间步骤的激活都依赖于先前步骤的激活。对于 MS-TCN 和 MS-TCN++，所有时间步骤的激活都是并行计算的。

5 结论

我们提出了两种用于时域动作分割任务的多阶段架构。当第一阶段生成初始预测时，该预测由更高的阶段迭代地改良。代替通常使用的时域池化，我们使用扩展卷积来增大接受域。实验评估证明了我们的体系结构在捕获动作类之间的时域相关性和减少过度分割错误方面的能力。我们进一步引入了平滑损失，这进一步提高了预测质量。我们还引入了一个双扩展层，它可以捕获局部和全局特征，从而提高性能。此外，我们表明，在改良阶段共享参数会产生一个效率更高的模型，但性能略有下降。在从不同视角记录的三个具有挑战性的数据集上，我们模型的表现优于目前最先进的方法。因为我们的模型是完全卷积的，所以它在训练和测试期间都非常高效和快速。

致谢

The work has been funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – GA 1927/4-1 (FOR 2535 Anticipating Human Behavior) and the ERC Starting Grant ARCA (677650).

参考文献

[1] S.-J. Li, Y. AbuFarha, Y. Liu, M.-M. Cheng, and J. Gall, "Ms-tcn++: Multi-stage temporal convolutional network for action segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2020.

[2] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Advances in Neural Information Processing Systems (NIPS)*, 2014, pp. 568–576.

[3] C. Feichtenhofer, A. Pinz, and R. Wildes, "Spatiotemporal residual networks for video action recognition," in *Advances in Neural Information Processing Systems (NIPS)*, 2016, pp. 3468–3476.

[4] J. Carreira and A. Zisserman, "Quo vadis, action recognition? A new model and the kinetics dataset," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 4724–4733.

[5] C. Feichtenhofer, H. Fan, J. Malik, and K. He, "Slowfast networks for video recognition," in *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[6] M. Rohrbach, S. Amin, M. Andriluka, and B. Schiele, "A database for fine grained activity detection of cooking activities," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012, pp. 1194–1201.

[7] S. Karaman, L. Seidenari, and A. Del Bimbo, "Fast saliency based pooling of fisher encoded dense trajectories," in *European Conference on Computer Vision (ECCV), THUMOS Workshop*, 2014.

[8] D. Oneata, J. Verbeek, and C. Schmid, "The lear submission at THUMOS 2014," *HAL01074442*, 2014.

[9] H. Kuehne, J. Gall, and T. Serre, "An end-to-end generative framework for video segmentation and recognition," in *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2016.

[10] C. Lea, A. Reiter, R. Vidal, and G. D. Hager, "Segmental spatiotemporal CNNs for fine-grained action segmentation," in *European Conference on Computer Vision (ECCV)*. Springer, 2016, pp. 36–52.

[11] A. Richard, H. Kuehne, and J. Gall, "Weakly supervised action learning with RNN based fine-to-coarse modeling," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[12] C. Lea, M. D. Flynn, R. Vidal, A. Reiter, and G. D. Hager, "Temporal convolutional networks for action segmentation and detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[13] P. Lei and S. Todorovic, "Temporal deformable residual networks for action segmentation in videos," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 6742–6751.

[14] L. Ding and C. Xu, "Weakly-supervised action segmentation with iterative soft boundary assignment," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 6508–6516.

[15] Y. Abu Farha and J. Gall, "MS-TCN: Multi-stage temporal convolutional network for action segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[16] S. Stein and S. J. McKenna, "Combining embedded accelerometers with computer vision for recognizing food preparation activities," in *ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 2013, pp. 729–738.

[17] A. Fathi, X. Ren, and J. M. Rehg, "Learning to recognize objects in egocentric activities," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011, pp. 3281–3288.

[18] H. Kuehne, A. Arslan, and T. Serre, "The language of actions: Recovering the syntax and semantics of goal-directed human activities," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014, pp. 780–787.

[19] A. Fathi and J. M. Rehg, "Modeling actions through state changes," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013, pp. 2579–2586.

[20] A. Fathi, A. Farhadi, and J. M. Rehg, "Understanding egocentric activities," in *IEEE International Conference on Computer Vision (ICCV)*, 2011, pp. 407–414.

[21] S. Bhattacharya, M. M. Kalayeh, R. Sukthankar, and M. Shah, "Recognition of complex events: Exploiting temporal dynamics between underlying concepts," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014, pp. 2243–2250.

[22] Y. Cheng, Q. Fan, S. Pankanti, and A. Choudhary, "Temporal sequence modeling for video event detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014, pp. 2227–2234.

[23] H. Kuehne, A. Richard, and J. Gall, "Weakly supervised learning of actions from transcripts," *Computer Vision and Image Understanding*, vol. 163, pp. 78–89, 2017.

[24] H. Kuehne, A. Richard, and J. Gall, "A Hybrid RNN-HMM Approach for Weakly Supervised Temporal Action Segmenta-

tion," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 04, pp. 765–779, 2020.

[25] K. Tang, L. Fei-Fei, and D. Koller, "Learning latent temporal structure for complex event detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012, pp. 1250–1257.

[26] N. N. Vo and A. F. Bobick, "From stochastic grammar to bayes network: Probabilistic parsing of complex activity," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014, pp. 2641–2648.

[27] A. Richard and J. Gall, "Temporal action detection using a statistical language model," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 3131–3140.

[28] B. Singh, T. K. Marks, M. Jones, O. Tuzel, and M. Shao, "A multi-stream bi-directional recurrent neural network for fine-grained action detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 1961–1970.

[29] S. Singh, C. Arora, and C. V. Jawahar, "First person action recognition using deep learned descriptors," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 2620–2628.

[30] A. Van Den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. W. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio." in *ISCA Speech Synthesis Workshop (SSW)*, 2016.

[31] S.-H. Gao, M.-M. Cheng, K. Zhao, X.-Y. Zhang, M.-H. Yang, and P. Torr, "Res2net: A new multi-scale backbone architecture," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[32] K.-N. C. Mac, D. Joshi, R. A. Yeh, J. Xiong, R. S. Feris, and M. N. Do, "Learning motion in feature space: Locally-consistent deformable convolution networks for fine-grained action detection," in *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[33] Z. Shou, D. Wang, and S.-F. Chang, "Temporal action localization in untrimmed videos via multi-stage CNNs," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[34] H. Xu, A. Das, and K. Saenko, "R-C3D: Region convolutional 3d network for temporal activity detection," in *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[35] Y. Zhao, Y. Xiong, L. Wang, Z. Wu, X. Tang, and D. Lin, "Temporal action detection with structured segment networks," in *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[36] Y.-W. Chao, S. Vijayanarasimhan, B. Seybold, D. A. Ross, J. Deng, and R. Sukthankar, "Rethinking the faster R-CNN architecture for temporal action localization," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[37] R. Zeng, W. Huang, M. Tan, Y. Rong, P. Zhao, J. Huang, and C. Gan, "Graph convolutional networks for temporal action localization," in *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[38] J. Gao, Z. Yang, K. Chen, C. Sun, and R. Nevatia, "Turn tap: Temporal unit regression network for temporal action proposals," in *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[39] S. Buch, V. Escorcia, C. Shen, B. Ghanem, and J. Carlos Niebles, "SST: Single-stream temporal action proposals," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[40] J. Gao, Z. Yang, and R. Nevatia, "Cascaded boundary regression for temporal action detection," in *British Machine Vision Conference (BMVC)*, 2017.

[41] T. Lin, X. Liu, X. Li, E. Ding, and S. Wen, "BMN: Boundary-matching network for temporal action proposal generation," in *IEEE International Conference on Computer Vision (ICCV)*, 2019, pp. 3889–3898.

[42] S. Yeung, O. Russakovsky, G. Mori, and L. Fei-Fei, "End-to-end learning of action detection from frame glimpses in videos," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[43] S. Buch, V. Escorcia, B. Ghanem, L. Fei-Fei, and J. C. Niebles, "End-to-end, single-stream temporal action detection in untrimmed videos," in *British Machine Vision Conference (BMVC)*, 2017.

[44] F. Long, T. Yao, Z. Qiu, X. Tian, J. Luo, and T. Mei, "Gaussian temporal awareness networks for action localization," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[45] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh, "Convolutional pose machines," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 4724–4732.

[46] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in *European Conference on Computer Vision (ECCV)*. Springer, 2016, pp. 483–499.

[47] M. Dantone, J. Gall, C. Leistner, and L. Van Gool, "Body Parts Dependent Joint Regressors for Human Pose Estimation in Still Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 11, pp. 2131–2143, 2014.

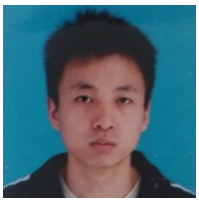
[48] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[49] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2015, pp. 234–241.

[50] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

Shijie Li received his Bachelor degree in Automation Engineering from University of Electronic Science and Technology of China in 2016 and his Master degree in computer science from the Nankai University in 2019. Since 2019, he is a PhD student at the University of Bonn. His research interests include action recognition and scene understanding.

Yazan Abu Farha received his Bachelor degree in computer systems engineering from Birzeit University in 2013 and his Master degree in computer science from the University of Bonn in 2017. Since 2018, he is a PhD student at the University of Bonn. His research interests include action recognition and anticipation.



Yun Liu is a PhD candidate at College of Computer Science, Nankai University. He received his bachelor degree from Nankai University in 2016. His research interests include computer vision and machine learning.

Ming-Ming Cheng received his PhD degree from Tsinghua University in 2012. Then he did 2 years research fellow, with Prof. Philip Torr in Oxford. He is now a professor at Nankai University, leading the Media Computing Lab. His research interests includes computer graphics, computer vision, and image processing. He received research awards including ACM China Rising Star Award, IBM Global SUR Award, and CCF-Intel Young Faculty Researcher Program. He is on the editorial boards of IEEE TIP.

Juergen Gall obtained his B.Sc. and his Masters degree in mathematics from the University of Wales Swansea (2004) and from the University of Mannheim (2005). In 2009, he obtained a Ph.D. in computer science from the Saarland University and the Max Planck Institut für Informatik. He was a postdoctoral researcher at the Computer Vision Laboratory, ETH Zurich, from 2009 until 2012 and senior research scientist at the Max Planck Institute for Intelligent Systems in Tübingen from 2012 until 2013. Since 2013, he is professor at the University of Bonn and head of the Computer Vision Group.