# Abstract

我们介绍了Siam R-CNN，这是一种孪生式重新检测体系结构，可释放的两阶段目标检测方法在视觉目标跟踪应用上的能力。 我们将Siam R-CNN 与新提出的tracklet-based dynamic编程算法相结合，该算法利用重新检测第一帧模板和先前帧预测的优势，对要跟踪的对象和潜在的干扰对象的完整历史进行建模。 这使我们的方法可以做出更好的跟踪决策，并可以在长时间遮挡后重新检测被跟踪的对象。 最后，我们提出了一种新颖的难例挖掘策略，以提高Siam RCNN对外观相似的对象的鲁棒性。 拟议的跟踪器在十个跟踪基准上均可以和当前的最佳性能相比拟，对于长期跟踪而言，其效果尤其明显。

# 1 Introduction

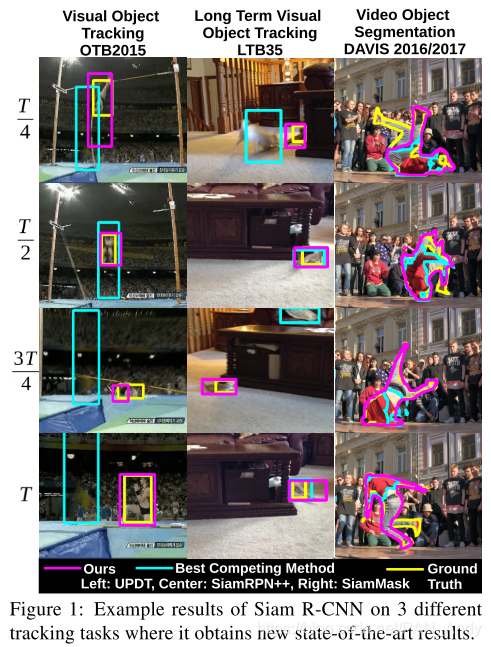
（本文基于的范式是利用重检测进行视频目标追踪。前期利用重检测进行视觉目标追踪的方法通常会受到相似干扰物的影响，其解决策略主要是利用先前的预测提供强位置先验信息和模型在线更新，但是这两种方法都避免不了模型的漂移问题。本文的工作主要有：（1）提出一种新的Siam R-CNN追踪器，利用孪生结构将Faster R-CNN应用于解决视觉目标追踪问题；（2）提出一种新的难例挖掘方法，能有效缓解干扰物对追踪结果的影响；（3）提出一种基于Tracklets的动态规划算法，能够在遮挡、目标消失等挑战中实现有效追踪。Siam R-CNN在六个短期追踪基准和四个长期追踪基准中均优于先前提出的方法。另外，利用现有的box-to-segmentation部件能够利用追踪框实现视频目标分割，在四个常见追踪数据集上比仅使用初始框标注进行分割的其他方法性能都好。）

我们使用“通过重新检测进行跟踪”的范式来处理视觉目标跟踪。 我们提出了一种功能强大的新型检测器Siam R-CNN，它是采用孪生结构的Faster R-CNN [74]的改编版，可通过确定区域提议的结果是否为模板区域的同一对象来重新检测图像中任何位置的模板对象，然后对该对象进行边界框回归。 我们的两阶段重新检测体系结构在对象尺寸和纵横比变化较大时具有很强的鲁棒性，因为区域提议被调整为相同的尺寸，这与流行的基于互相关的追踪方法形成了对比[49]。

通过重新检测进行跟踪已有很长的历史，可追溯到Avidan [1]和Grabner等人的开创性工作。 [28]。 由于存在与模板对象非常相似的干扰对象，重新检测具有挑战性。 过去，解决干扰物的问题主要通过先前的预测[4、49、48]中的强空间先验或通过在线适应[1、28、2、76、30、77、42]来解决。 这两种策略都容易产生漂移。

相反，我们通过在Siam R-CNN重新检测器设计之外做出两个新的贡献来解决干扰物体。首先，我们介绍一种新颖的难例挖掘程序，该程序专门针对困难的干扰物训练我们的重新检测器。其次，我们提出了一种新颖的Tracklet动态规划算法（TDPA），该算法通过重新检测前一帧中的所有候选对象框，并随时间将框分组为tracklets（短目标追踪），从而同时跟踪所有潜在对象（包括干扰对象）。然后，它使用动态编程，根据视频中所有目标对象和干扰对象tracklets的完整历史记录，在当前时间步中选择最佳对象。通过对所有潜在物体的运动和相互作用进行显式建模，并从分组为tracklets的检测中收集相似信息，Siam R-CNN能够有效执行长期跟踪，同时抵抗跟踪器漂移，并能够立即重新检测消失后的物体。我们的TDPA在每个时间步中只需要进行少量新的重新检测，即可在线迭代更新跟踪历史。这使Siam R-CNN在使用ResNet-101骨干网时可以以4.7帧/秒（FPS）的速度运行，而在使用ResNet-50骨干网时以超过15 FPS的速度运行，输入提议越少，输入图像的越小。

我们提供了大量数据集的评估结果。 Siam R-CNN在六个短期跟踪基准（OTB2015 [99]，TrackingNet [66]，GOT-10k [38]，NFS [43]，VOT2015 [46]和OTB50 [99]）上优于所有以前的方法。 在四个长期跟踪基准上，LTB35 [62]，UAV20L [65]，LaSOT [23]和OxUVA [86]上Siam R-CNN取得了特别出色的结果，比以前的方法高出10个百分点。 通过使用现成的box-to-segmentation网络获取分割掩膜，Siam R-CNN在DAVIS 2017（包括val和test-dev）[72]、YouTube-VOS 2018 [101]和DAVIS 2016 [71]上还优于所有以前的仅使用第一帧边界框（不带掩膜）的视频目标分割方法 。 所有代码和模型都将可用。



# 2 Related Work

Visual Object Tracking (VOT). VOT是在给定对象第一帧边界框的情况下完成视频目标追踪任务。 VOT通常根据基准进行评估，例如OTB [98，99]，年度VOT挑战[47，45]，以及更多[66，38，117，65，43]。 最近，已经提出了许多长期跟踪基准[62、86、23]，这些基准将VOT扩展到了更加困难和真实的设置，其中必须在许多帧上跟踪对象，并且对象会消失并重新出现。

许多经典方法使用在线学习的分类器来重新检测整个图像上的感兴趣对象[1、28、2、76、30、77、42]。 相反，Siam R-CNN通过离线训练来学习预期的外观变化而不是在线学习分类器。

像我们的Siam R-CNN一样，许多最新方法都使用Siamese体系结构来实现VOT。 孪生区域提议网络（SiamRPN [49]）使用单阶RPN [74]检测器，该检测器通过将深层模板特征与当前帧的深层特征互相关来重新检测模板目标。 在这里，单阶段意味着直接对锚定框anchor[57]进行分类，这与两阶段体系结构[74]相反，后者首先生成提案proposals，然后对齐其特征并在第二阶段对其进行分类。 最近的跟踪方法对SiamRPN进行了改进，使其能够应对相似物的干扰（DaSiamRPN [118]），或者添加了级联（C-RPN [25]），或者使用更深的网络架构（SiamRPN + [114]和SiamRPN ++ [ 48]），或者维护一组多样化的目标模板（THOR [78]）。 这些（还有更多[7，35，63]）仅在先前预测的小窗口内搜索对象。 DiMP [5]在遵循了这种范例的基础上使用元学习来学习鲁棒的目标和背景外观模型。

VOT的其他最新发展包括使用在线学习的特定域层[67]，学习自适应空间滤波器正则化[17]，利用类别特定的语义信息[85]，使用连续[20]或分解[18]卷积以及实现精确的卷积，使用重叠预测网络的边界框预测[19]。Huang等学者 [39]提出了一种将任意一种检测器转换成跟踪器的框架。 像SiamR-CNN一样，它们也应用两阶段体系结构，但是它们的方法依赖于元学习，并且精度要低得多。

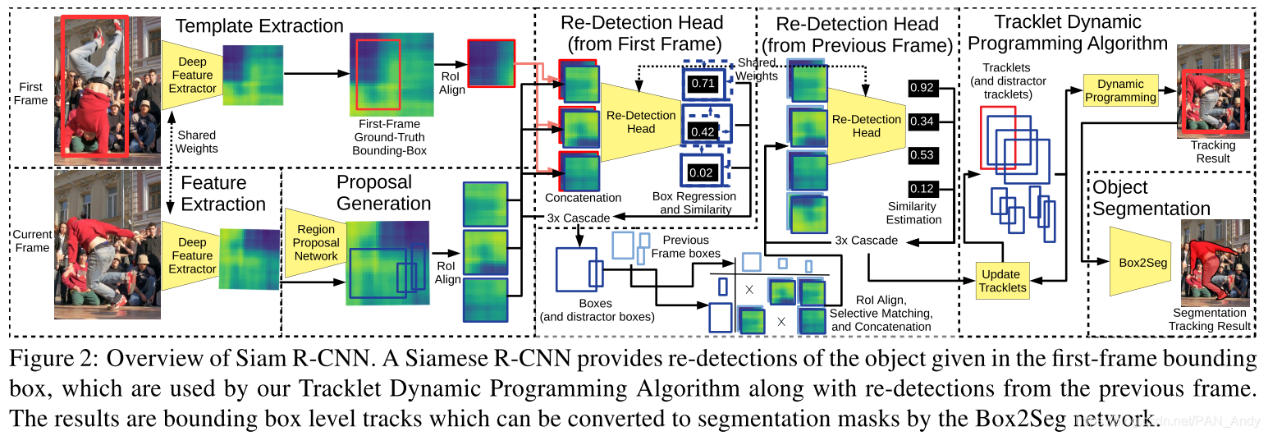
当检测置信度较低时，主要通过扩大这些孪生跟踪器的搜索窗口来解决长期跟踪问题[118，48]。 相比之下，我们使用两阶孪生检测器来搜索整个图像，从而在许多基准测试中产生比当前方法更好的结果，尤其是对于长期跟踪而言。

Video Object Segmentation (VOS). VOS是VOT的扩展，其中提供了一组模板分割掩膜，并且需要在每个帧中生成分割掩膜。 许多方法都在模板掩膜的基础上进行微调[8、64、89、52、3、60]，这种方法可以产生非常好的分割结果，但是速度很慢。 最近，有几种方法在没有微调的情况下使用了第一帧掩膜[12、105、13、37、100、101、87、69]，运行速度更快，但通常效果不佳。

有零星的几个的方法[94，108]解决仅在使用给定模板边界框而不使用掩膜的情况下产生掩膜跟踪结果的难题。 我们调整了在此设置中调整Siam R-CNN来解决VOS的任务，通过使用第二个网络为框跟踪结果生成掩膜。

# 3 Distractor-aware Siamese Networks

受孪生跟踪器[45，99，47]成功的启发，我们将孪生架构用于重新检测器。 许多最新的跟踪器[118、94、48、49、5]采用单阶检测器架构。 对于单图像物体检测的任务，已显示出诸如Faster R-CNN [74]之类的两阶检测器网络优于单阶检测器。 受此启发，我们将跟踪器设计为孪生两阶检测网络。 第二阶段可以通过串联感兴趣区域（RoI）对齐的特征，直接将提议的感兴趣区域（RoI）与模板区域进行比较。 通过将提议区域和参考图像对齐到相同的尺寸，Siam RCNN可以对对象尺寸和纵横比变化保持鲁棒性，而使用流行的互相关操作很难做到这一点[49]。 图2显示了包括Tracklet动态规划算法（TDPA）的Siam R-CNN的概述。



## 3.1 Siam R-CNN

Siam R-CNN是基于两阶检测架构的孪生重检测器。 具体来说，我们采用了在COCO [56]数据集上经过预训练的用于检测80个对象类别的Faster R-CNN网络。 该网络由骨干特征提取器和随后的两个检测阶段组成。 首先是与类别无关的RPN，然后是用于特定于类别的检测头。 我们固定骨干网和RPN的权重，并用我们的重检测头替换类别特定的检测头。

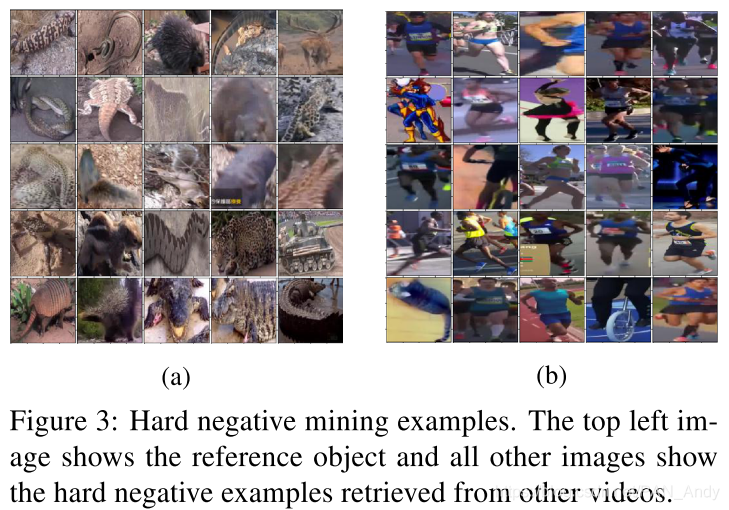
我们通过执行RoI Align [33]为该建议区域提取深层特征，为RPN建议的每个区域的重检测头创建输入特征。我们还在第一帧中获取了初始化边界框的RoI Aligned深层特征，然后将它们串联在一起，并将合并的特征馈入1×1卷积中，这将特征通道的数量减少了一半。 然后将这些聚合特征送入带两个输出类别的重检测头中：拟议的区域要么是参考目标，要么不是。 我们的检测头使用三级级联[9]，没有共享权重。 重新检测头的结构与Faster R-CNN的检测头的结构相同，不同之处在于仅使用两个输出类，并且通过串联创建重检测头的输入特征。 将主干和RPN参数冻结，并且使用视频数据集中的成对帧仅对重检测头（连接后）进行跟踪训练。 在这里，将一帧中的对象用作参考，并训练网络来重新检测另一帧中的同一对象。

## 3.2 Video Hard Example Mining

在常规的Faster R-CNN训练期间，从目标图像中RPN提议的区域中采样第二阶段的负训练样本。 但是，在许多图像中，只有很少的相关负样本。 为了最大程度地提高重检测头的判别能力，我们需要在严格的负样本上进行训练。 在以前的工作中已经探索了进行检测的难例挖掘（例如[26，79]）。 但是，与寻找用于检测的通用难例挖掘方法不同，我们通过从其他视频中检索参考对象来构建困难训练样本。

Embedding Network. （这个Embedding Network网络其实就是一个特征提取网络，提取ground truth bbox图像区域的特征，用这个特征代表这块图像对象。因为提取的是代表特征，所以它应该具有同一对象的特征距离会很近，而不同对象的特征之间的距离会很远。） 为当前视频建构难例的一个直观方法是选择相关的视频，这些视频里有和当前目标相同的类别[118]。 但是，对象类别标签并不总是可用，并且同一类的某些对象可能易于区分，而不同类的某些对象也可能是难例。 因此，我们建议使用embedding network，它是受行人重识别启发的，该网络为该对象外观的每个地面真相边界框提取一个嵌入向量来代表这个对象。 我们使用来自PReMVOS [60]的网络，该网络使用batch-hard triplet loss[36]，先在COCO的各个类别上进行训练，然后再在YouTube-VOS上进行训练以消除各个对象实例之间的歧义。 例如，在嵌入空间中，两个不同的对象应该相距较远，而不同帧中的同一对象之间距离应相近。

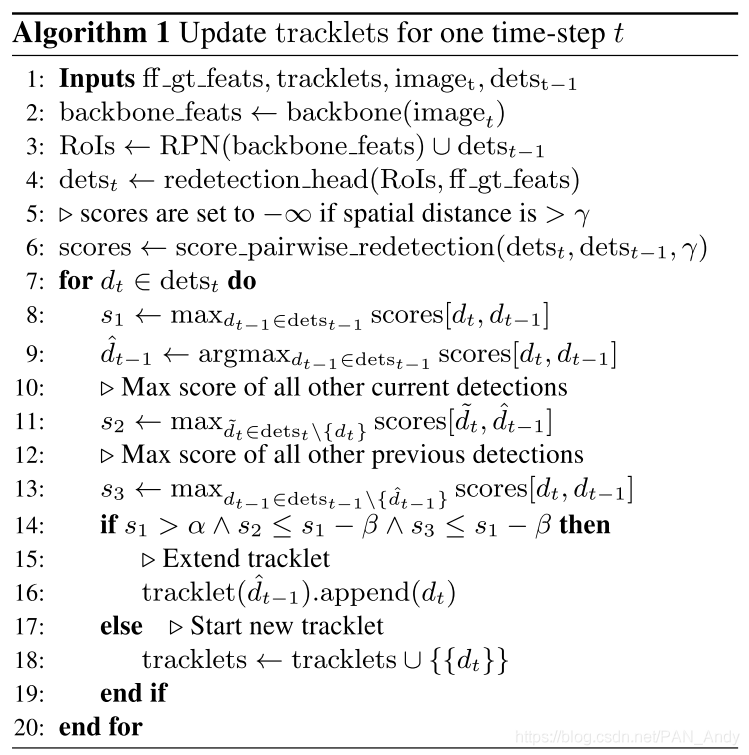
Index Structure. 接下来，我们构建了类似最近邻查询结构，创建了有效的索引结构，并使用它在嵌入空间中找到被跟踪对象的最近邻 （可能是被追踪对象的 困难样例） 。 图3示出了所检索的难例负例的例子。 可以看出，大多数负例都是非常相关和很难差别。



Training Procedure. （Embedding Network 和 Index Structure 都是为都是为了Training Procedure中提供更高效的训练难例提取而设计的。大致的思路就是，先提取每个真值追踪框的ROI aligned特征，然后在训练的过程中，由Index Structure生成选取的追踪对象的难例，完成训练。这里的难例与追踪对象是非常相似的，这种训练方法应该是可以提高网络对同类的不同实例的差别能力。） 从其他视频中在线评估主干网络来检索当前视频帧的难例是非常昂贵的。 取而代之的是，我们为训练数据的每个地面真实框预先计算了与RoI对齐的特征。 对于每个训练步骤，通常会选择一个随机视频和该视频中的对象，然后选择一个随机参考图像和一个随机目标帧。 然后，我们使用索引结构从其他视频中为参考框检索10,000个最近邻的边界框，并从其中采样100个作为难例。 可以在补充材料中找到有关视频难例挖掘的更多详细信息。

## 3.3 Tracklet Dynamic Programming Algorithm

我们的Tracklet动态规划算法（TDPA）隐式跟踪感兴趣的对象和潜在的干扰对象，因此干扰对象可以始终被抑制。 为此，TDPA维护一组tracklets，就是一个短的检测序列（short sequences of detections），几乎可以确定tracklets中的检测序列是属于同一对象的。 然后，它使用基于得分的动态规划算法为第一帧和当前帧中的模板对象选择tracklets中最有可能的序列。 （所以，TDPA选择的是tracklets中得分最高的一个检测序列） 每个检测都由边界框、重检测得分及其RoI对齐特征进行定义。 此外，每个检测都是一个tracklets的一部分。 一个tracklet具有开始时间和结束时间，它由一组检测定义，从开始到结束时间的每个时间步都进行一次检测。也就是，一个tracklet中不允许有时间间隔。



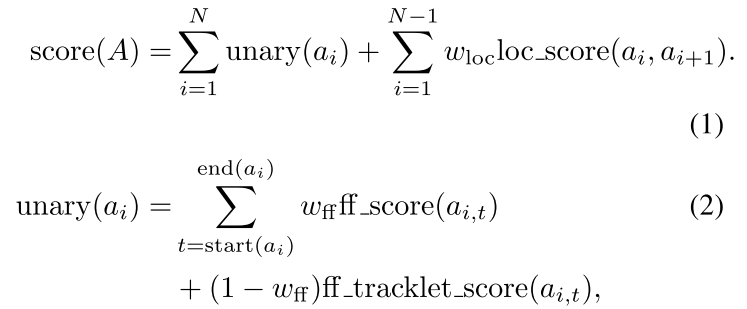
Tracklet Building. 我们为第一帧地面真实边界框提取ROI对齐特征（ff\_gt\_feat），并初始化仅由该框组成的tracklet。对于每个新帧，我们按以下方式更新tracklet set（参见算法1）：我们提取当前帧的骨干特征，并评估区域提议网络（RPN）以获取感兴趣的区域（RoI，第2–3行）。为了补偿潜在的RPN误报，RoI的集合将由前一帧的边界框输出扩展。 （应该是利用前一帧预测的bbox） 我们在这些RoI上运行重新检测头（包括边界框回归），以生成一组对第一帧模板的重新检测（第4行）。 之后，我们在当前的检测上重新运行重检测头的分类部分（第6行），但是这次以来自前一帧的检测 作为参考，而不是第一帧地面真实框， 计算每对检测之间的相似性得分（分数）。

为了测量两次检测的空间距离 （两次检测的空间距离，在追踪中的含义就是，前一帧的追踪框bbox和当前帧的追踪框bbox之间的距离），我们用其中心坐标x和y表示其边界框，并用宽度w和高度h表示它们的宽度，其中x和w用图像宽度标准化，而y和h用图像高度标准化，所有值都在0到1之间。然后，通过范数给出两个边界框（x 1，y 1，w 1，h 1）（x1，y1，w1，h1）（x1，y1，w1，h1）和（x 2， y 2， w 2，h 2）（x2，y2，w2，h2）（x2，y2，w2，h2）之间的空间距离，即。为了节省计算量并避免错误匹配，我们仅针对空间距离小于γ的检测对计算成对相似度得分，否则将相似度得分设置为。

当与新检测的相似性得分高（）并且没有歧义时，我们通过当前帧检测来扩展前一帧的tracklet（第7-20行）。例如没有其他检测具有与该tracklet相同强度的相似性（小于β），并且没有其他tracklet与该检测具有几乎相同的相似度（小于β）。 只要有任何歧义，我们都会启动一个新的tracklet，它最初包含一个检测。 然后在tracklet评分步骤中消除歧义。 （tracklet的扩张和新建的过程大致就是，当当前帧中可以明确检测到追踪目标时，就把这个检测加入到tracklet中，如果当前帧的检测结果有可能不是待追踪目标，那么就新建那一个tracklet。就是在启动新tracklet的过程中，会自动隐式地追踪到相似干扰物，但是通过动态规划求解，可以求解出待追踪目标的tracklet序列。）

（a是一个tracklet,tracklet是一个检测序列，A代表一个追踪，也是一个由tracklet组成的序列）

Scoring. 一个追踪 )是由N个不重叠的tracklets组成的序列，例如，其中start和end表示tracklet的开始和结束时间。 （这个式子表明，tracklet 的结束时间，肯定在下一个tracklet 的开始时间之前。这也表明tracklet是一组时间上连续的检测，在长期追踪中它可以理解为，在目标的一个存在期间内，每帧检测结果（目标追踪框）的集合。而在长时追踪中，目标可以有多个存在期间，将这多个存在期间的tracklet组合起来，就是在整个视频中对这个目标的追踪A AA了） 总得分包含一个度量单独tracklets质量的一元得分和惩罚tracklets之间空间跳跃的位置得分，例如：



其中ff\_score代表时间t tt时的tracklet 的检测 重检测置信度 （也就还是检测置信度得分detection score），它来自于使用第一帧真值框作为参考的重检测头。总会有一个tracklet包含第一帧地面真值边界框，我们将其称为第一帧tracklet 。一个tracklet中的所有检测detections，都有非常高的机会成为这个tracklet中的初始检测的正确继续 （是初始检测的一个追踪） ，因为有歧义的tracklets会被终止 。因此，对第一帧tracklet的最新检测也是几乎可以肯定是正确对象的最新观察结果。因此，我们还将最近检测到的第一帧tracklet用作重新检测的附加参考信息。这个得分表示为ff\_tracklet\_score，和ff\_score进行线性组合。

  两个tracklet ​和​之间的位置分数由最后一次检测​的边界框（x，y，w，h）（x，y，w，h）（x，y，w，h）与第一次检测​的边界框之差的负L1范数给出，即：

在这里插入图片描述

Online Dynamic Programming. 我们通过维护一个数组θ来有效地找到具有最高总得分（等式1）的tracklet序列。数组θ中每个tracklet a，存储从第一帧tracklet开始的到以a为结尾的最优tracklet序列的总得分。

一旦一个tracklet不被扩展了，它将被终止。 因此，对于每个新帧，仅需要重新计算已经扩展或新创建的tracklet的得分。对于一个新的时间步（一个新的追踪任务？新的视频？），首先，我们将第一帧tracklet 设置为=0。之后，对于每个已经更新或新创建的tracklet a，计算方式如下: 表示的应该是 是一个在a之前的tracklet，这个max求的是在a之前的tracklet中，得分的最大值。整个动态规划方程就是，等于tracklet a的重检测质量得分unary(a)加上tracklet a之前的tracklet得分的最大值，再加上 和a之间的加权位置得分）

θ [ a ] = unary ⁡ ( a ) + max ⁡ a ~ : end ⁡ ( a ~ ) < start ⁡ ( a ) θ [ a ~ ] + w loc ⁡ loc ⁡ − score ⁡ ( a ~ , a ) \theta[a]=\operatorname{unary}(a)+\max \_{\tilde{a}: \operatorname{end}(\tilde{a})<\operatorname{start}(a)} \theta[\tilde{a}]+w\_{\operatorname{loc}} \operatorname{loc}\_{-}\operatorname{score}(\tilde{a}, a)

θ[a]=unary(a)+

a

~

:end(

a

~

)<start(a)

max

​

θ[

a

~

]+w

loc

​

loc

−

​

score(

a

~

,a)

为了在很长的序列中保持效率，我们允许两个tracklet之间的最大时间间隔为1500帧，这对于大多数应用来说足够长。

  在当前帧中更新θ θθ之后，我们选择动态规划得分最高的tracklet a ^ \hat{a}

a

^

，即a ^ ＝ arg ⁡ max ⁡ a θ [ a ] \hat a ＝ \arg \max \_a \theta[a]

a

^

＝argmax

a

​

θ[a]。 如果选定的tracklet a ^ \hat a

a

^

在当前帧中不包含一个检测，则我们的算法表明该对象不存在 （目标出视野或被完全遮挡） 。 对于需要在每个帧中进行预测的基准，我们使用所选tracklet的最新框，并将其分配为0。

## 3.4 Box2Seg

  为了为VOS任务生成分割掩膜，我们使用了来自PReMVOS [60]的现成的边界框到分割（Box2Seg）网络。 Box2Seg是具有Xception-65 [16]主干的全卷积DeepLabV3 + [11]网络。 它已经在Mapillary [68]和COCO [56]上受过训练，可以为边界框输出掩膜。 Box2Seg速度很快，在跟踪之后运行它只需每帧每个对象0.025秒。 我们组合重叠的掩膜，以使像素较少的掩膜位于顶部。

## 3.5 Training Details

  Siam R-CNN建立在Faster R-CNN [74]实现和[96]的预训练权重基础之上，具有ResNet-101-FPN主干[34，55]，组归一化[97]和级联[9]。 它已经在COCO [56]上从零开始[32]进行了预训练。 除非另有说明，否则我们将同时在多个跟踪数据集的训练集上训练Siam R-CNN：ImageNet VID [75]（4000个视频），YouTube-VOS 2018 [101]（3471个视频），GOT-10k [38]（ 9335个视频）和LaSOT [23]（1120个视频）。 我们使用运动模糊和灰度数据增强[118]以及伽玛和比例数据增强进行训练。

# 4 Experiments

  我们在标准视觉对象跟踪、长期跟踪以及VOS基准上评估Siam R-CNN。 我们为DAVIS 2017训练集上的Tracklet动态编程算法（参见第3.3节）调整了一组超参数，因为我们没有用它来训练重新检测器的训练集。 我们在所有基准上使用这些超参数来评估结果，而不是针对每个基准单独调整参数。

## 4.1 Short-TermVisual Object Tracking Evaluation

  对于短期VOT，我们使用以下基准：OTB2015 [99]，VOT2018 [45]，UAV123 [65]，TrackingNet [66]和GOT-10k [38]。 在补充材料中，我们还评估了VOT2015 [46]，VOT2016 [44]，TC128 [54]，OTB2013 [98]和OTB50 [99]。

  OTB2015. 我们评估OTB2015（100个视频，平均590帧的平均长度），计算重叠阈值变化时的成功率和精度。 方法通过成功曲线下的面积（AUC）进行排名。 图4将我们的结果与8个最先进的（SOTA）跟踪器进行了比较[6、48、18、5、67、20、19、118]。 Siam R-CNN达到了70.1％的AUC，这等于UPDT先前的最佳结果[6]。

  UAV123. 图5显示了我们在UAV123 [65]（123个视频，平均915帧的平均长度）上的结果，与六个SOTA方法[5、19、48、118、6、18]相比，其度量标准与OTB2015相同。 我们实现了64.9％的AUC，接近DiMP-50 [5]先前的最佳结果65.4％。

  NfS. 表1显示了我们在NfS数据集上的结果[43]（30FPS，100个视频，平均479帧），与5种SOTA方法相比。 Siam R-CNN的成功分数为63.9％，比DiMP-50先前的最佳成绩高1.9个百分点[5]。

  TrackingNet. 表2显示了我们在TrackingNet测试集上的结果[66]（511个视频，平均长度442帧），与五种SOTA方法相比。 Siam R-CNN的成功分数为81.2％，比DiMP-50的先前最佳成绩高7.2个百分点[5]。 就精度而言，差距超过10个百分点。

  GOT-10k. 图6显示了我们在GOT-10k [38]测试集（180个视频，平均127帧的平均长度）上与六种SOTA方法相比的结果[5、116、19、90、48、18]。 在此基准上，方法仅允许使用GOT-10k训练集作为用于训练的视频数据。 因此，我们从COCO预训练开始训练新模型，并且仅在GOT-10k上训练。 我们实现了64.9％的成功率，比DiMP-50先前的最佳结果高3.8个百分点[5]。 这表明Siam R-CNN相对于所有以前方法的优势不仅是由于训练数据不同，还在于跟踪方法本身。

  VOT2018. 表3显示了我们在VOT2018 [45]上的结果（60个视频，平均长度356帧）。按照VOT2018的标准，使用基于重置的评估，一旦丢失对象，跟踪器将在五帧后用地面真值框重新启动，并受到处罚。主要评估标准是预期平均重叠（EAO）[46]。这种极端的短期跟踪方案不是为带有TDPA的Siam R-CNN设计的。它通常会触发重置，通常（没有基于重置的评估）Siam R-CNN可以自动恢复，从而得出EAO为0.140。由于VOT2018是重要的跟踪基准，因此我们创建了Siam R-CNN的简单短期版本，它假定前一帧的预测是正确的（必须是正确的，否则它将被重置），然后取平均重新检测的预测第一帧参考，并重新检测先前的预测，并将其与强大的空间先验结合在一起（补充材料中有更多详细信息）。我们的短期Siam R-CNN的EAO得分为0.408，可与许多SOTA方法相比。值得注意的是，两个版本的Siam R-CNN均获得最高的准确性得分。最后一行显示使用普通（非重置）评估时的平均重叠（AO）。在此设置中，Siam R-CNN达到0.476 AO，接近SiamRPN ++ 0.498的最佳结果。

## 4.2 Long-Term Visual Object Tracking Evaluation

  为了评估Siam R-CNN进行长期跟踪的能力，我们评估了三个基准，即LTB35 [62]，LaSOT [23]和OxUvA [86]。 在补充材料中，我们还评估了UAV20L [65]。 在长期跟踪中，序列会更长，并且对象可能会消失并再次出现（LTB35每个视频平均消失12.4个，每个平均消失40.6帧）。 在所有这些基准测试中，Siam R-CNN的性能明显优于所有以前的方法，这表明我们通过重新检测方法进行跟踪的优势。 通过在整个图像上进行全局搜索，而不是在先前预测的局部窗口内进行搜索，我们的方法具有更好的抗漂移性，并且可以在消失后轻松地重新检测目标。（这是如何实现的，怎样达到这种效果的，需要重点关注一下。）

  LTB35. 图7显示了与其他八种SOTA方法相比，我们的方法在LTB35基准（也称为VOT18-LT）上的结果[62]（35个视频，平均长度为4200帧）。 要求跟踪器输出目标的置信度，以便在每个帧中进行预测。为一定范围的置信度阈值计算Precision(Pr)和Recall(Re)。F-score的计算为F = 2 P r R e P r + R e F=\frac{2PrRe}{Pr+Re}F=

Pr+Re

2PrRe

​

。追踪器按在所有阈值上的F-score最大值进行排序。我们比较了2018年VOT-LT挑战赛中表现最好的6种方法[45]和SiamRPN ++ [48]和SPLT [104]。 Siam RCNN的F得分为66.8％，优于以前的所有方法，比以前的最佳结果高3.9个百分点。

  LaSOT. 图8显示了与9种SOTA方法[5、19、48、67、80、4、112、29、18]相比，LaSOT测试集[23]（280个视频，平均2448帧的平均长度）上的结果。 Siam RCNN以64.8％的成功率和72.2％的归一化精度获得了空前的成果。 与以前的最佳方法相比，成功率高8个百分点，归一化精度高7.4个百分点。

  OxUvA. 表4显示了与五种SOTA方法相比，OxUvA测试集[86]（166个视频，平均长度3293帧）的结果。 跟踪器必须在每个帧中做出艰难的决定，即对象是否存在。 为此，我们将检测器的置信度与在dev set上调整的阈值进行比较。 方法按真实阳性率（TPR）和真实阴性率（TNR）的最大几何平均值（MaxGM）进行排名。 Siam R-CNN的MaxGM比所有以前的方法高出10个百分点。

## 4.3 Video Object Segmentation (VOS) Evaluation

  我们在VOS基准DAVIS17 [72]和YouTube-VOS [101]上进一步评估了我们的方法。除了其他跟踪评估之外，这还评估了两个方面。首先，能够产生准确的细分跟踪结果（在运行Siam R-CNN进行跟踪之后，我们引入了Box2Seg）。其次，要求同时跟踪多个对象。在VOS中，跟踪器使用J \mathcal{J}J度量（预测的和地面真理掩码的平均联合相交（IoU））和F \mathcal{F}F度量（边界相似性）进行评估。我们还引入了一个新的度量J b o x \mathcal{J}\_{box}J

box

​

，它是J \mathcal{J}J度量，但使用围绕预测的和地面真实掩膜的边界框进行计算，可用于直接测量跟踪质量而无需掩膜。我们将VOS方法分为不使用第一帧掩膜的方法（如我们的方法），直接使用第一帧掩膜的方法以及对其进行微调的方法。除了带有Box2Seg的标准Siam R-CNN，我们还展示了在第一帧掩膜上对Box2Seg进行300步微调的结果（Siam RCNN从未进行微调，仅使用第一帧标注框）。

  DAVIS 2017. 表5和图9显示了与SOTA方法相比，DAVIS 2017验证集的结果（72个视频（30个视频，平均长度67.4帧，每个视频平均2.03个对象））。方法按J \mathcal{J}J和F \mathcal{ F}F的平均值排序。SiamR-CNN明显优于仅使用第一帧边界框SiamMask [94]的以前的最佳方法，提高了14.8个百分点。为了评估Box2Seg有多少改进以及我们的跟踪有多少改进，我们将Box2Seg应用于SiamMask的输出。这确实改善了结果，但仍比我们的方法差7.3个百分点。我们还使用Box2Seg运行SiamRPN ++ [48]和DiMP-50 [5]进行比较。作为跟踪器可实现的性能的参考，我们在真值标注框上运行Box2Seg，结果得分为82.6％。

  即使不使用第一帧掩膜，Siam R-CNN的性能也比使用第一帧掩膜的许多方法（例如RGMP [100]和VideoMatch [37]）要好，甚至比执行缓慢的第一帧微调的方法要好（例如OSVOS-S [ 64]和OnAVOS [89]。我们的方法也更实用，因为与边界框初始化相比，手工创建完美的第一帧分割掩膜要繁琐得多。如果可以使用第一帧掩膜，那么我们就可以对此进行微调，以丧失速度为代价将结果提高4.2个百分点。我们在补充材料中报告了对DAVIS17 test-dev基准和DAVIS16 [71]的进一步评估。

  YouTube-VOS. 表6和图10显示了我们的方法在YouTube-VOS 2018基准测试[101]上的结果（474个视频，平均长度26.6帧，平均每个视频1.89个对象），并和六种SOTA方法相比。 评估区分了训练集中（出现）和未训练（未出现）中的类别。 方法以总分O \mathcal{O}O进行排名，总分O \mathcal{O}O是可见和不可见类的J \mathcal{J}J和F \mathcal{F}F指标的平均值。 Siam R-CNN再次优于所有不使用第一帧掩膜的方法（提高了15.5个百分点），并且也优于PReMVOS [60，59]和除STM-VOS [69]以外的所有其他先前方法。 使用经过微调的Box2Seg的方法版本将结果提高了4.9个百分点。

## 4.4 Ablation and Timing Analysis

  表7显示了在三个数据集上的Siam R-CNN消融实验及其运行速度（使用V100 GPU）。 Siam R-CNN使用ResNet-101主干网以每秒4.7帧（FPS）的速度运行，每帧1000 RPN提议，以及TDPA。

  “No hard ex”行。 显示了不进行难例挖掘的结果（参见第3.2节）。 难例挖掘可将所有数据集的结果提高多达1.7个百分点。

  我们将TDPA与仅在每个帧中使用最高评分的重新检测（“ Argmax”）和我们用于基于复位的VOT18评估的短期算法（“Short-term”）进行比较。 TDPA在所有数据集中均胜过这两个。 Argmax稍快一些，因为它不会使用先前的预测作为参考来进行重新检测。 对于长期数据集，Argmax明显优于Short-term甚至所有以前的方法，从而突出了通过重新检测进行长期跟踪的跟踪强度。

## 4.5 Making Siam R-CNN Even Faster

  表7还显示了旨在提高Siam R-CNN速度的三项改进的结果（较小的主干网络，较小的输入分辨率和较少的RoI提议）。使用ResNet-50骨干网络进行评估时，Siam RCNN的执行速度稍快，但仍可达到SOTA结果（在LaSOT上为62.3，而在相同骨干网络下的DiMP-50则为56.8）。这表明追踪性能的提升不仅是由于骨干网络较大，而且还归因于我们通过重新检测方法进行的跟踪。

  我们还评估了将图像输入尺寸减小为较小的图像边缘长度，而不是800（而不是800（第"1 2 \frac{1}{2}

2

1

​

res"行））。 这还会导致两个基准的性能仅略有下降，而OTB2015的性能则略有提高。

  “ 100 RoIs”行显示了使用RPN中的100 RoI而不是1000的结果。这几乎使速度加倍，因为大多数计算都发生在重新检测头中。 这样只会导致两个基准上的分数下降很少，而LTB35的结果有所改善。 这表明，即使Siam RCNN基于两阶段检测架构，它也可以非常快速地运行，因为只需要很少的RoI。

  所有这三个速度改进（ResNet-50 + 100 RoIs +1 2 \frac{1}{2}

2

1

​

res。）的最快设置，使用V100 GPU可以达到每秒15.2帧，但仍然可以获得出色的效果，特别是对于长期跟踪应用场景中。 使用ResNet-101骨干网络而不是ResNet-50（100 RoIs +1 2 \frac{1}{2}

2

1

​

res）的相同设置以每秒13.6帧的速度运行，与标准Siam R-CNN相比，在这三个数据集上可实现出色的结果，并且最多损失1.6个百分点，而运行速度几乎快三倍。

## 4.6 Generic Object Tracking Analysis

  Siam R-CNN应该能够跟踪任何通用对象。但是，它的主干和RPN仅在COCO中的80个对象类上进行了训练，然后在进行重新检测训练时被固定。在图11中，与其余的方法相比，我们调查了OTB2015序列中包含非COCO对象的44％的RPN调用情况。默认情况下有1000个提案，RPN对未知对象的召回率仅为69.1％，而已知对象的召回率则为98.2％。一种解决方案是增加使用的提案数量。当使用10 000个建议时，RPN对未知对象的召回率达到98.7％，但会使Siam R-CNN运行得慢得多（大约1 FPS）。我们的解决方案是改为包含先前帧重检测（最多100个）作为其他建议。当使用1000个RPN建议时，这将未知对象的召回率提高到95.5％，甚至在仅使用100个RPN建议时，也会导致91.4％的足够的未知对象召回率。这说明了为什么即使几乎一半的对象不是来自COCO类，Siam R-CNN能够胜过OTB2015上所有以前的方法。我们还对DAVIS17和LTB35数据集进行了召回分析，其中几乎所有对象都属于COCO类，并且我们实现了出色的召回（请参见右图11）。

# 5 Conclusions

  我们将Siam R-CNN引入具有Tracklet 动态规划算法的孪生两阶段全图像重检测架构。 Siam R-CNN在十个跟踪基准上均优于以前的所有方法，对于长期跟踪而言，结果尤其出色。 我们希望我们的工作会激发以后使用两阶段体系结构和全图像重新检测进行跟踪的工作。