Leveraging Long-Range Temporal Relationships Between Proposals for Video Object Detection

Mykhailo Shvets
UNC at Chapel Hill
mshvets@cs.unc.edu

Wei Liu
Nuro Inc
w@nuro.ai

Alexander C. Berg
UNC at Chapel Hill
aberg@cs.unc.edu

Abstract

Single-frame object detectors perform well on videos sometimes, even without temporal context. However, challenges such as occlusion, motion blur, and rare poses of objects are hard to resolve without temporal awareness. Thus, there is a strong need to improve video object detection by considering long-range temporal dependencies. In this paper, we present a light-weight modification to a single-frame detector that accounts for arbitrary long dependencies in a video. It improves the accuracy of a single-frame detector significantly with negligible compute overhead. The key component of our approach is a novel temporal relation module, operating on object proposals, that learns the similarities between proposals from different frames and selects proposals from past and/or future to support current proposals. Our final "causal" model, without any offline post-processing steps, runs at a similar speed as a single-frame detector and achieves state-of-the-art video object detection on ImageNet VID dataset.

1. Introduction

Modern single-frame detectors [5, 13, 14, 15] sometimes perform well on the task of object detection in video even without any temporal information. However, some challenges still exist which a single-frame detector cannot resolve without looking at temporal context. Those include occlusion, motion blur, rare poses of object, etc. Thus, it is a natural desire to modify the single-frame detector to consider information from more than one frame. Recently, many methods try to use image level information from nearby frames to help detection. It is important to note that such methods usually assume that the scene and objects do not move drastically. For instance, FGFA [26] and MANet [19] use optical flow from nearby images, STSN [1] applies deformable convolutions, and D&T [3] computes dense correlation between frames. Performance of the key components of these methods (flow field, deformation field, and correlation, respectively) degrades with time dramatically, making it hard to leverage the relations between frames that are far apart in time. In those methods, long-term dependencies are only considered with extra offline post-processing steps, such as SeqNMS [4]. Moreover, even for the online mode of those methods, there is a significant degrade of speed. For example, FGFA and MANet report $2.5 - 3$ times slowdown compared to a single-frame detector. To speed up, many other methods either use a lighter backbone network such as MobileNet [8] or run a single-frame detector only at key frames and propagate image-level features using optical flow [7, 25]. These approaches usually suffer in accuracy because information is not directly computed from each frame. By leveraging long-range temporal relationships between proposals from different frames in a video, we show that we can include long-term dependencies even without any post-processing steps, while running online at a similar speed as a single-frame detector and still achieve state-of-the-art accuracy on ImageNet VID dataset.

Our approach is based on aggregating features from related parts of a video in order to make a better decision about whether a putative detection is in fact correct. This involves determining which parts of a video are related to a potential detection, weighted averaging of the features for those parts, and the final detection decision on those updated new features. Our approach is trained end to end, so the features are learned in order to facilitate both identifying related parts and to perform detection after related features are averaged together. We consider region proposals (from FPN detector [12]) as both potential detections and as potential related parts of a video. It is also possible to use detector outputs, e.g. from a fast high-recall single-stage detector. In any case, a significant challenge for considering long-range temporal relationships is that there are many possibilities. One key aspect of our approach is that we decompose the problem of which proposals should support detections in a frame into computations on small sets (2-3) of frames, and aggregate the results.

The main component of our method is a novel proposal-

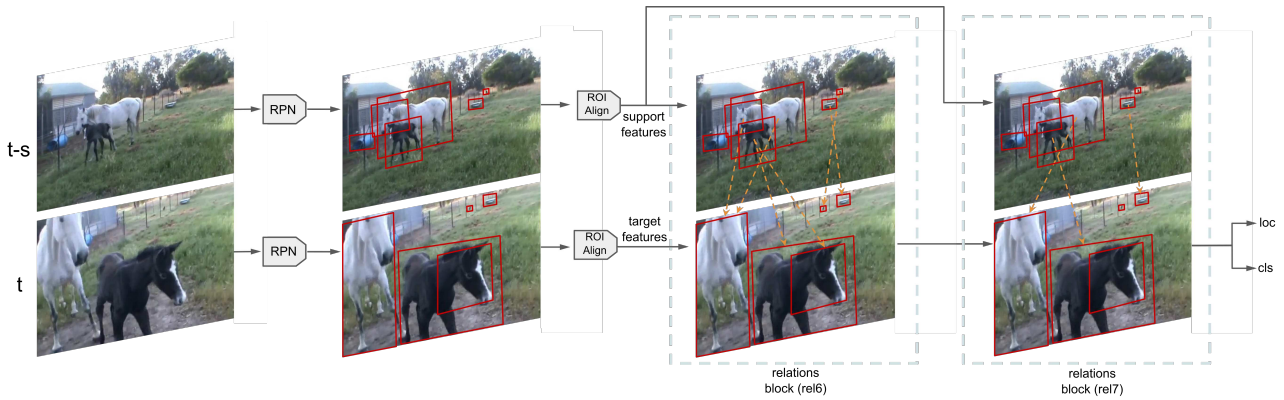

Figure 1: Overall architecture. A single-frame two-stage detector is modified to include the relation module that updates the features for proposals in the target frame based on those of proposals in the support frames. For simplicity we show single support frame $t - s$, which corresponds to the *causal* mode with temporal kernel $(K = 2, s)$ where s is the stride. N and M proposals are selected for the second stage by RPN from target and support frames, respectively. After features are extracted with ROIAlign, we inject two relation blocks into the prediction head to inform the target frame of support instances. During inference, the temporal kernel is efficiently applied to multiple strides, gathering rich long-range temporal support.

level temporal relation block that updates the features for potential detections (proposals) in a target frame by modeling appearance relationships between proposals from target and support frames. This module draws its inspiration from self-attention mechanisms [18] that have recently been proven to be beneficial for single-frame recognition [9, 20]. In contrast, we show how to use the relation module for modeling *inter*-frame dependencies on object proposals, and introduce direct supervision to *learn* the appearance affinity. Moreover, we demonstrate that our approach benefits from accumulating long-range temporal relations, while competing models degrade when attempting to account for frames more than a fraction of a second apart [1, 3]. Part of this advantage comes from learning what proposals may be related to a potential detection based on appearance features but ignoring location and temporal disparity.

We summarize our contribution as follows:

- a novel proposal-level temporal relation block that learns appearance similarities, and enriches target features using features from support frames;
- a method of applying the relation block to incorporate long-term dependencies from multiple support frames in a video;
- online inference with comparable speed as a baseline single-frame detector;
- thorough experiments of relation graph construction methods, including new feature normalization and adding graph supervision to learning.

2. Related work

Single-frame object detection. Single-frame detectors have been improved dramatically in both accuracy and speed in the last few years. However these methods lack temporal awareness to resolve hard cases in videos, such as occlusion, motion blur, and rare object poses, etc. In our work, we build on top of a two-stage detector (FPN [12]), modify the instance-level detection branch by infusing appearance proximity to proposals from many support frames and improve the instance features, which yields confident prediction for the challenging cases in video.

Pixel-level features across frames. There has been a line of research that augments the single-frame detector with temporal connections *using pixel-level features*. D&T [3] builds a dense correlation map between two feature maps of consecutive video frames, and exploits instance track ids to learn bounding box frame-to-frame motion. FGFA [26] uses optical flow to guide the feature propagation, and achieves pixel-to-pixel alignment of features between frames with proper warping. STSN [1] avoids explicit optical flow computation, but uses deformable convolution. More generally, [20] introduces non-local networks to completely drop the locality of estimated correspondences. All of these methods use the relationships of pixel-level features, which are known to degrade quickly with time. Indeed, FGFA uses 21 consecutive frames for feature aggregation, thus looking around for only a fraction of a second. Spatial stride of more than 2 frames reduces the accuracy of STSN. Similarly, D&T has a performance drop when modeling relationships between frames that are 10 frames apart rather than immediately consecu-

tive frames. In contrast to these methods, we learn to relate fine-grained *instance-level appearance features* which do not degrade with time, and deliberately ignore spatial location of instances to unleash the use of larger temporal strides and aggregation over multiple temporal strides. Our module operates on a few hundreds of proposals per frame, which allows feature aggregation with the historical information, and can efficiently back-propagate information down to the RPN and backbone network end-to-end.

Instance-level features across frames. Modeling relations between objects has become a growing line of research [9, 17], drawing inspiration from attention mechanism and graph convolution. Recently, [21] has shown that modeling instance-level feature relationships is important to achieve state-of-the-art performance for action recognition. It uses graph convolution with a learnable graph representation to create a descriptive feature for each clip. Similarly, we use a learnable graph for modeling appearance proximity. However, our task is more fine-grained because we perform per-instance prediction of object location and class in each frame, while [21] predicts only class for the video clip. Long-term instance-level dependencies have been proven highly important in [22], which is modified from attention-based non-local block [20]. Our Frozen Support Setting is inspired by this work, and a lighter model demonstrated improvement from using additional frozen network. However, the final model shows no performance boost with this approach, while being over 2x slower. Thus, while [22] uses three different networks to succeed in the task of action recognition, we can achieve state-of-the-art performance in video object detection by extracting the features for both target and support instances *within a single network*.

Long-term relations are also studied in [23] via spatial-temporal memory alignment in a recurrent fashion. Support propagation is done on pixel level, based on local neighbourhood, while our instance-level approach assumes no notion of locality. [23] reportedly has difficulties with recognizing fast-moving objects due to challenge of local appearance feature propagation, but our method is robust to fast motion according to motion-specific breakdown below.

As mentioned above, the non-local block [20] is originally proposed for pixel-level aggregation. A similar mechanism is applied for instance features in [9] for object detection in a single frame through the use of multiple relation heads. Our relation module has the same placement in architecture. In contrast, we sample supporting instances in different frames across the temporal dimension to aggregate long-term relational dependencies. Thus, while [9] improves instance features from surrounding context within the frame, we aim at boosting detection confidence based on other occurrences of the same instance in video. Apart from switching to the level of instances, we also find that performing feature normalization before computing the appearance proximity is important for long-range connections. Moreover, we add a supervision loss directly on the graph, which favours high similarity between detections of the same instance.

Post-processing. Current state-of-the-art methods for video object detection all leverage pixel-level features [1, 3, 19, 26]. As we have mentioned, those methods only include relatively short-term dependencies, and their performance degrades over longer time period. To improve the accuracy and account for long-term dependencies, offline post-processing technique such as SeqNMS [4] is required to link and to re-score the proposals, where appearance similarity is not considered. In our model, we explicitly consider the appearance similarity deep into the network by allowing rich feature support through back-propagation.

3. Method

In this section we present the temporal relation block with object proposals, describe its integration in a single-frame detector, and also introduce our training and inference settings. We define a *target frame* as a frame where final prediction is done at the moment. The target frame is allowed to have multiple *support frames*, which are used for strengthening the current proposals.

Our goal is to learn to update instance-level features, and consequently predictions of the detector while running detection in an *online fashion*, without any post-processing steps. In order to achieve that, we learn appearance-based relations between object proposals.

Appearance-based relation block. Assuming there are N target and M support proposals with features from D-dimensional space ($X_{\text{target}} \in \mathbf{R}^{N \times D}$ and $X_{\text{support}} \in \mathbf{R}^{M \times D}$), we construct an attention mechanism to update the target features with the attention-weighted average of the support features. We will index target instances with index i, and support instances with index j, and thus $\mathbf{x}_i$ denotes row i for matrix X_{target}, and $\mathbf{x}_j$ — row j for matrix X_{support}. The overview of the relation block is presented in Figure 2. First, both target and support features are embedded with a linear layer. Second, both matrices undergo feature normalization (FeatNorm in the figure), which is different from previous works where appearance relations are built as a direct correlation of the embedded features [9, 20, 22]. We notice that otherwise the correlation is dominated by feature magnitudes rather than the actual relations (i.e. if both instance i and j are of high magnitude, their relation is high). This is especially important in ROI-level features in detection, as those are learned to be biased and have high magnitude. We experiment with unit-normalized and zero-centered unit normalized vectors, where the correlation in fact turns into Pearson uncentered and centered correlation coefficients. The third step is to construct the appearance relations matrix G ($\times$ in the figure) by perform-

ing simple correlation between all pairs of target features $\mathbf{x}_i$ and support features $\mathbf{x}_j$:

$$G = G(X_{\text{target}}, X_{\text{support}}) : \mathbf{R}^{N\times D} \times \mathbf{R}^{M\times D} \rightarrow \mathbf{R}^{N\times M} \tag{1}$$

Because matrix G is used further for computing attention weights on the support proposals, we notice that good properties of G include having high values $G_{i,j}$ for proposals $\mathbf{x}_i$ and $\mathbf{x}_j$ that correspond to the same instance in the video, and low values $G_{i,j}$ for unrelated instances. Indeed, if instance $\mathbf{x}_i$ is presented in the supporting frames, the corresponding proposals should be linked and reinforce it. In order to enforce that constraint, we add a supervision loss to the graph G (GraphLoss). The motivation described above best fits into the notion of contrastive loss, which encourages small distance between same instances, and large distance between different instances. $\mathcal{L}(\mathbf{x}_i, \mathbf{x}_j, y_{i,j})$ is

$$\frac{1-y_{i,j}}{2} d(\mathbf{x}_i, \mathbf{x}_j)^2 + \frac{y_{i,j}}{2} \left[max\left(0, \mu - d(\mathbf{x}_i, \mathbf{x}_j)\right)\right]^2, \tag{2}$$

where $y_{i,j}$ is a dissimilarity label (i.e., $y_{i,j} = 1$ if i and j are different instances and 0 otherwise). Notice that due to feature normalization, $\|\mathbf{x}_i\|_2 = \|\mathbf{x}_j\|_2 = 1$, and in the case when graph is correlation of feature pairs, there is a direct relation between G and d generated by L_2 scalar product:

$$d^2(\mathbf{x}_i, \mathbf{x}_j) = \|\mathbf{x}_i\|_2^2 + \|\mathbf{x}_j\|_2^2 - 2(\mathbf{x}_i, \mathbf{x}_j) = 2\left(1 - G_{i,j}\right) \tag{3}$$

After the descriptive appearance relations matrix G is constructed, it is used as a weight in the attention mechanism for support features. For that, rows of G are normalized with softmax, forming $\hat{G}$. Due to feature normalization, original values of G are bounded, so we use multiplicative constant of 10 before the softmax. One can imagine that if a target instance i has a strong relations support from proposals $j_1, j_2, \ldots, j_k$, then the values of $G_{ij_1}, G_{ij_2}, \ldots, G_{ij_k}$ are high, and only those will "survive" after the softmax operation. On the other hand, if i has no corresponding proposals in the supporting frames, then all values in the i-th row will be low, and the softmax will not favour any of the supporting instances, thus the target instance will receive simple average support that carries no relational signal. $\hat{G}$ is used to aggregate the embedded support features with a matrix multiplication layer ($\times$ in Figure 2), which after an additional linear layer embedding, form a final aggregated support feature matrix of the same size as the input target feature matrix (the output dimensionality of the last linear layer is designed to be exactly D). Thus, we use element-wise sum ($+$) to update the target features.

Overall, the block is similar to the non-local block in [20], but with a few differences. First, it is applied to instances rather than to pixel-level features. Second, target and support features come from different frames, and are even allowed to be computed from independent models, as discussed in the frozen support model setting below. Third, we add feature normalization, which aids descriptive graph construction, and allows effective supervision on G.

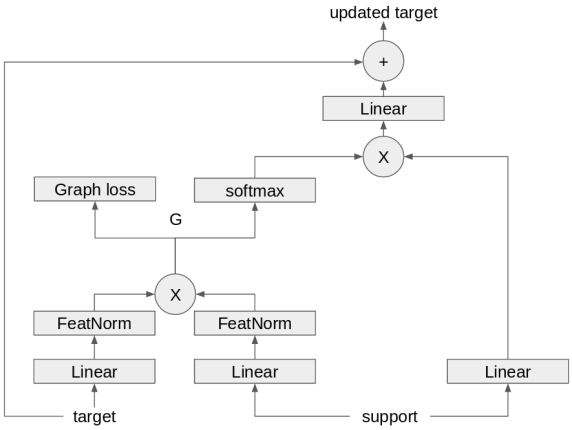

Figure 2: Relation block: the core component in the new model. It accepts $N \times D$ target and $M \times D$ support feture matrices. Both tensors are embedded with linear layers, and passed through feature normalization layer (unit normalization, or zero-centered unit normalization). Pairwise correlations of the normalized features (via their matrix multiplication $\times$) constitute $N \times M$ graph matrix G. This matrix is supervised, given matched track ids of the proposals to favour high values for same instances, and low values for different. The graph is further normalized row-wise to create an attention distribution of M support over N target proposals. This attention matrix is used as a weight for embedded support features (bottom-right linear layer) in matrix multiplication operator ($\times$). After another transformation, target features are augmented with element-wise sum $+$

Until now, we have only discussed how to update target features based on support proposals. While the target frame is simply the frame of interest at the moment, there are several ways of constructing the set of support frames during both training and inference, and computing their features.

Causal and Symmetric modes. To pick supporting frames for a given target frame, we introduce a notion of temporal kernel. *Temporal kernel* is a tuple (K, s), denoting size and stride. The size K is a number of frames that is included in relation reasoning. Thus, for one selected target frame there are $K-1$ supporting frames. The stride s restricts the K frames to be sampled at a uniform time spacing of s frames.

We also define two kernel modes for a given target frame t. In *Causal* mode the supporting frames are chosen only from the past: those are $t-s, t-2s, \ldots, t-(K-1)s$. In *Symmetric* mode, the supporting frames are chosen from

both the past and the future: $t - \lfloor K/2 \rfloor s, \ldots, t - s, t + s, \ldots, t + \lfloor K/2 \rfloor s$ (the kernel size K is assumed to be odd).

Single Model and Frozen Support Model settings. Computing target and support features within a single model is an attractive choice due to high efficiency. In our *Single Model* setting both target and support features are the ROI features pooled from the same layer and the network trained end-to-end. However, there are two potential issues. First, when kernel size is set to K, during training time at least K frames are processed on one device. Given large image resolution for object detection (images are resized to a shorter side of 600 pixels), it is only possible to fit a few images in the memory of a standard GPU, thus the kernel is limited in size. Second, support feature distribution is changing during training of the detector, as it is computed from the same set of parameters as target features. To address those, we introduce a Frozen Support Model setting, where a single-frame pre-trained detector (the frozen support model) is used to extract instance level support features, while the main detector runs single image at a time, also receiving the fixed support features as additional input. Notice that during training the Frozen Support Model does not need to allocate large memory, as no backpropagation is required, which enables larger kernel size K during training. In our experiment section we demonstrate that in fact learning both features from the same model is beneficial, and our aggregation over different strides levels down the need for large kernel.

Training and inference. As discussed in Section 2, geometric relations between instances are believed to degrade with time, even if sophisticated motion propagation methods are used [1, 3, 26]. In our model, geometric relations between boxes are intentionally not used in order to enable capturing long-term dependencies during inference. We demonstrate the possibility to apply the kernel with large temporal stride s to capture long-term relations. To that end, we train a relation module to be *agnostic to the temporal kernel stride* s, which enables aggregation across different temporal strides during inference. Thus, kernel size K is fixed as a parameter to a model for both training and testing, but the temporal stride is chosen randomly on the fly. During *training*, s is chosen once. Notice that the model is trained with single s due to memory limitations, which were mentioned above, but the relation module is agnostic to stride during inference due to random sampling. This enables *aggregation over multiple strides*. During *inference*, the kernel is applied multiple times for several different strides, and feature aggregation is performed. We show that extending the temporal stride (we use up to $s = 256$) not only does not degrade the performance, but shows consistent improvement of the accuracy.

4. Experiments

Dataset. We carry out our main experiments on the ImageNet VID dataset [16]. The training set consists of 3862 video snippets with a total of over a million frames (1122397). The frames are fully annotated with bounding boxes across 30 object categories. Associated ground truth track id is used in our contrastive loss to determine positive and negative pairs during training. There are 555 validations snippets with a total of 176126 frames. As [1, 3, 10, 11, 26], we train on the intersection of the ImageNet VID and DET sets (using 30 VID classes, the subset of 200 DET classes). As videos are of varying length, it is important to balance the number of frames selected as targets not to overfit to long videos. The same reasoning applies to the class imbalance in DET. For fair comparison, we sample our target frames from the set publicly released by [26]. At most 15 frames are subsampled from each video, and DET:VID balance is approximately 1:1.

Detector settings. We build our system on top of the single-frame FPN detector [12] with ResNet [6] backbones. In our setting, Feature Pyramid is only used in the Region Proposal Network, while ROI feature pooling uses only C_4 output (also known as ResNet's conv4 block). Such a modification is made in order to make sure that features come from the same distribution is Equation 1, avoiding the situation when features for i and j are pooled from different layers of the pyramid. The default IoU threshold of 0.7 is applied to RPN proposals. Unless otherwise noted, we use ResNet-50 backbone. The final models are reported with ResNet-101 and ResNeXt32x8d-101 [24]. We follow the established protocol of using the backbone model pretrained on ImageNet classification dataset. This makes a fair comparison with the state-of-the-art methods. We have also experimented with pretraining our full system on COCO dataset. Although COCO consists from still images, we can sample the same image K times to mimic the video frames flow. With that setting, our ResNet-50 causal model delivers 78.9 mAP, comared to 78.4 mAP as reported in Table 3.

Training. Input images are always resized to have a shorter side of 600 pixels. The system is trained on 4 GPUs, with each GPU holding 1 sample. Learning rate starts from 0.0025, and drops by a factor of 10 on iterations 80K and 120K. Iterations stop at 135K. During training, a sample consists of K frames (the size of the temporal kernel), including target frame t, and $K - 1$ support frames. Kernel stride s is sampled randomly, as our kernel is not only location-independent, but also temporal stride-agnostic.

In the Single Model setting, for all K frames the backbone extracts features, and the RPN yields the proposals. The bounding box head is run differently for support and target instances. Support proposals are extracted in the evaluation mode. That is, the proposals are not sub-sampled, and ground truth boxes are not included. We do this in or-

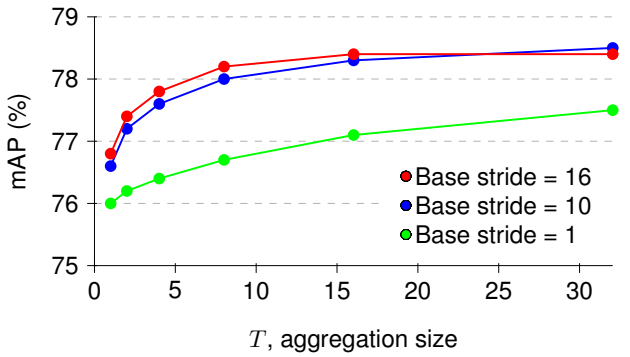

Figure 3: Accuracy improvement from aggregation over multiple strides. Using base stride $\tilde{s}$ means applying the same kernel with strides $\tilde{s}, \ldots, T\tilde{s}$, followed by average aggregation. Here $K = 2$, and Causal mode is chosen.

der to match the support distribution between training and inference modes. Indeed, during inference, targets are not known, so balanced sub-sampling is not possible. Back-propagation is still performed through the support features.

In the Frozen Support Model setting, $K - 1$ support frames are fed into the frozen model that yields the proposals and their features. The main model is run only for the target frame, while the support features are directly fed into the relation module. This setting is faster in training for larger kernels, but needs to run two models for inference, thus at least 2x slower than the baseline in the end.

Inference with feature aggregation across multiple strides. During inference we run the model once for the frame that is currently fed as an input. The relation module stores a buffer of features that it has seen so far in the current video, truncated at the maximum allowed stride. We apply the kernel multiple times during inference with different strides s. To that end, we introduce base stride $\tilde{s}$, which is used in the experiments. We say that kernel is applied T times with base stride $\tilde{s}$ when it is applied with strides $s \in \{\tilde{s}, 2\tilde{s}, \ldots, T\tilde{s}\}$. Thus, for the kernel with size K the total number of supporting frames is $(K-1)T$. In the Causal mode, support features come from previous frames that are stored in the buffer, and output for the target frame is given immediately. In the Symmetric mode, the output lags behind the input by $\lfloor K/2 \rfloor \max(s)$ frames. That is, when frame t arrives, it is stored in the buffer, and target frame $t - \lfloor K/2 \rfloor \max(s)$ is obtained from the buffer. When the video ends, the buffer releases all the remaining frames for final predictions. We pad edges of the video, repeating first and last frames to get support for all frames.

The features are simply averaged over the T outputs of the relation module. The effect of this operation is shown in Figure 3 for base strides $\tilde{s} \in \{1, 10, 16\}$ in Causal mode, where $K = 2$. There are two important things that the plot shows. First, a consistent improvement is gained from including more support frames, which saturates after $T = 16$ for larger $\tilde{s}$. This result is in line with [1, 19, 26]. Second, **increasing temporal stride delivers better performance**. The intuition behind this lies in a simple fact that immediate consecutive frames usually share the detection challenges (blur, occlusion, etc.), so looking at a larger temporal window, where detection is potentially easier, is beneficial. Unlike D&T [3] and STSN [1], our model is capable of such long-term aggregation, because no spatial consistency is assumed, and instances support each other based solely on appearance proximity. Based on the study that is shown in Figure 3, we choose $\tilde{s} = 16$ and $T = 16$ for our Single Model experiments that use $K = 2$ in causal mode and $K = 3$ in symmetric mode. In order the frames not to repeat, frozen support model uses strides $s \in [10, 21, 32, 43]$.

Uncentered dim=1024	Uncentered dim=10	Centered dim=10	Centered $_{\text{no loss}}$ dim=10
77.5	77.7	**78.4**	77.4

Table 1: Feature normalization. dim output dimensions are used in the two linear layers in the bottom-left of Figure 2 for graph construction.

fc6→fc6	fc6→fc7	fc6→{fc6, fc7}	fc7→fc7
77.3	77.0	**78.4**	77.5

Table 2: Various placements of relation blocks in a Single Model setting. $A \rightarrow B$ notation means that support features are taken as an output of layer A, while target features are from the layer B (A and B are $fc6$ or $fc7$ layers of the prediction head. Notice that $fc6 \rightarrow \{fc6, fc7\}$ includes two relation modules).

baseline	causal $K = 2$	symmetric $K = 3$	frozen support $K = 11$
72.4	**78.4**	**78.7**	**78.8**

Table 3: Causal, Symmetric modes, and Frozen Support model setting against ResNet-50 single-frame baseline. In causal mode, $K - 1$ supporting frames are taken from the past, while in Symmetric mode $\lfloor K/2 \rfloor$ frames are taken from both the past and the future. Frozen Support Model setting is in Symmetric mode and strides $[10, 21, 32, 43]$ are used for aggregation to avoid frame repetitions.

Ablation studies. In Section 3 we argue that feature normalization in Figure 2 is needed to learn meaningful correlations. Moreover, feature normalization allows us to apply the graph regularization loss. In Table 1 we experiment with uncentered and zero-centered normalizations. Our zero-centered normalization uses LayerNorm that includes additional learnable parameters. We also train a model without

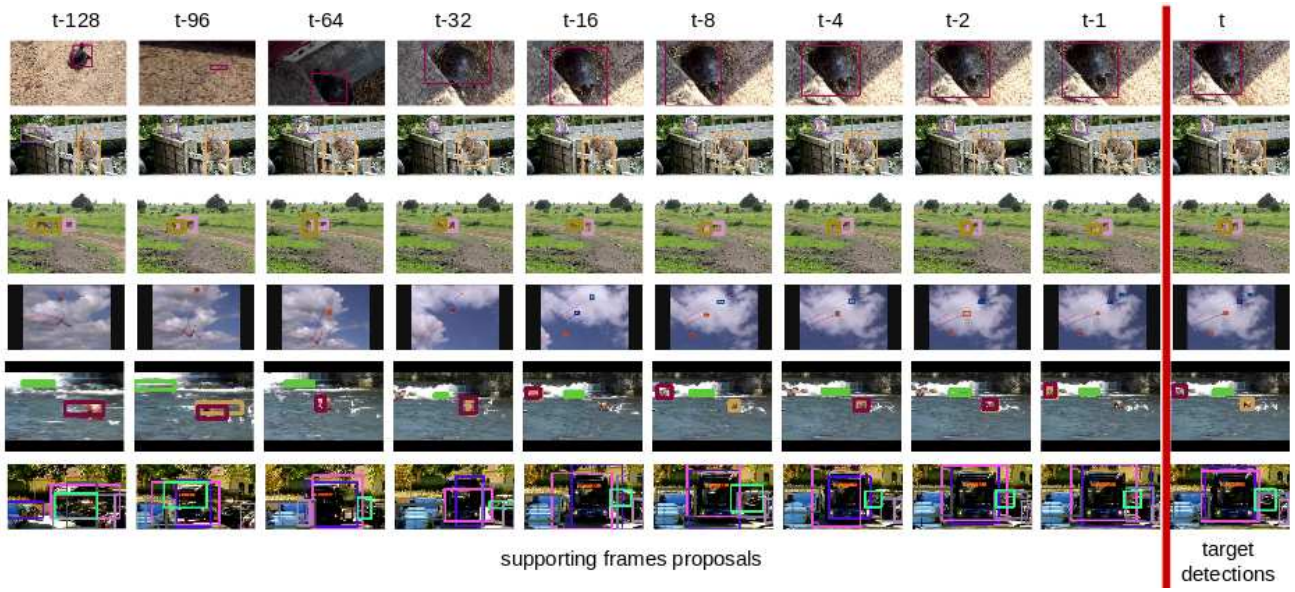

Figure 4: Support visualization. Shown is Causal mode with $K = 2$, where temporal kernel is applied multiple times to the fixed target image (timestep t, on the right) and different support frames $t - s$. We put the final prediction bounding boxes in frame t and **best supporting proposal** in each of the supporting frames. Instances are color-coded. Thus, the cat in the second row is supported by proposals centered at the same cat, and the same is true for the squirrel. Moreover, in the third row detected planes have different best support, which hints that our model learns to relate instances rather than classes, otherwise each plane could have support from *a single plane*, instead of *the plane closest in appearance*.

applying loss on the graph.

We notice that the features for the relation module can come from any instance-level layer that follows ROIAlign operation. Moreover, as in [9], our layer can be included multiple times in the instance prediction head. Table 2 shows different placements of the relation module. Header indicates which layer servers as a source for support feature, and which – for target (support source layer $\rightarrow$ target source layer(s). In one case, the relation module is applied twice (support from $fc6$ to $fc6$ and from $fc6$ to $fc7$). This configuration proves the best results and is used in our further studies, and in the final model. It is also shown in Figure 1. As in [9], we also tried using multi-head aggregation inside our relation block, but did not observe any performance boost. Thus, we avoid additional complexity and claim that *multi-stride long-term feature aggregation* (Figure 3) is more important. We compare our ResNet-50 based models to the single-frame baseline in Table 3.

Support visualization. For insight into which instances are learned to support putative detections in a target frame, we show Figure 4. The temporal kernel is applied multiple times in causal mode with $K = 2$ to the target image (timestep t, on the right) and different support frames $t - s$. Note that target instances receive support from the objects that are close in appearance, as encouraged by the graph loss. For example, in the second row, the cat is correctly linked to itself in the support frames, as is the squirrel. Also, the planes in the fourth row are supported by the planes with the same color of the tail.

Final results. Our final models are based on ResNet-101. The first part of Table 4 shows performance for models, including ours, with no post-processing. All our models show significant improvement over the competitors, and this holds even when heavy post-processing is applied to the competitors. That is SeqNMS that iteratively finds the best scoring tube in the video, suppresses other boxes, adjusts the scores, and repeats until no boxes are left for all but D&T. Similarly, D&T iteratively uses Viterbi algorithm to link propagated boxes into tubes and adjust scores.

In Table 4 we report 80.6 mAP, and the speed of 10 FPS on Titan X Pascal GPU for the causal model with only one relation block at $fc7$ layer. Our single-frame baseline runs at $14FPS$. In the case of ResNet-101 backbone, improvement from using the second block (default setting, as in Figure 1) is minor, and the final score is 80.7 mAP. In comparison, MANet that is also reasonably fast, reports 78.1 mAP while running at 5 FPS on the same GPU type. Moreover, MANet uses 12 nearby frames, which include *future frames*. This implies an additional delay to the online video stream.

We simplify our block to match the design of Non-Local

Model	mAP
FGFA [26]	76.3
STSN [1]	78.9
MANet [19]	78.1
D&T loss [3]	75.8
Ours, baseline	75.6
Ours, non-local block	79.0
Ours, causal $_{fc_7}$	**80.6**
Ours, causal	**80.7**
Ours, symmetric	**81.0**
Ours, frozen support	**80.6**
FGFA [26]+SeqNMS [4]	78.4
STSN [1]+SeqNMS [4]	80.4
MANet [19]+SeqNMS [4]	80.3
D&T [3] ($\tau = 10$)	78.6
D&T [3] ($\tau = 1$)	79.8
FGFA [26]+ [4] (Inception-ResNet)	80.1
D&T [3] (Inception-v4)	82.0
Ours, causal (ResNeXt-101)	**83.1**
Ours, symmetric (ResNeXt-101)	**84.1**

Table 4: Comparison to state-of-the-art methods. Except the last four rows, ResNet-101 backbone is used in all models. First part of the table shows the competing models without post-processing. Second part represents the boost that competing models get from heavy offline post-processing. *All our models outperform the competing models without any additional techniques.* The last part shows stronger backbone architectures, which are reported for the final best result of each method (all ours have no post-processing).

block [20] by removing feature normalization, graph loss, and using 512 embedding size. Other settings are the same (the block is applied to instance features with long-term aggregation). We achieve 79.0 mAP with this. We argue that *most performance gain comes from long-term aggregation*, as shown in Figure 3. But our novel components to the Non-Local block are important to get the extra **+1.7 mAP** gain.

The symmetric model is reported with $K = 3$. The kernel includes one frame from the past, and one from the future, and is applied 16 times, resulting in a total of 32 supporting frames, and performing at 81.0 mAP. Both causal and symmetric modes use the base stride of $\tilde{s} = 16$. Thus, the covered temporal range is up to 10 seconds, given that frames are extracted at $25 - 30$ FPS in ImageNet VID.

Additionally, we train ResNeXt32x8d-101 [24] backbone to compare with the best results reported in [26, 3] that are also obtained by switching to a better backbone. Our network outperforms those in causal mode with $K = 2$, demonstrating 83.1 mAP with no post-processing involved. The best result is reported for symmetric mode at 84.1 mAP.

Following the protocol outlined in [26], we provide a performance breakdown into slow, medium, and fast objects in Table 5. All objects are divided into these three categories based on their average IoU score between the corresponding instances across nearby frames. Thus, objects with score > 0.9 are slow, the ones in $[0.7, 0.9]$ are medium, and others are fast. Our baseline has 75.6 mAP. So, our causal mode delivers **+5.1 mAP**, where *most improvement comes from fast-moving objects* (**+9.3 mAP**). Indeed, fast motion is the most challenging case for video object detection. Our model is inherently robust to arbitrary motion pattern (i.e. relation block has no notion of geometric relationship).

	Slow	Medium	Fast	mAP
Ours, baseline	83.6	73.7	53.4	75.6
Ours, non-local block	84.6	78.1	59.1	79.0
Ours, causal	86.3	79.3	62.7	80.7
Ours, symmetric	86.7	79.5	64.2	81.0
Ours, frozen support	84.5	79.2	63.9	80.6

Table 5: Performance breakdown into slow-, medium-, and fast-moving objects for our ResNet-101 based models.

	$IoU_{0.05}$	$IoU_{0.5}$	$IoU_{0.75}$
Ours, baseline	49.9	36.7	11.0
Ours, causal	52.9	39.4	12.0

Table 6: Performance on EPIC KITCHENS dataset.

Additional experiments. While most previous works report on ImageNet VID dataset only, we add experiments on EPIC-KITCHENS [2]. The dataset consists of 272 cooking videos gathered by 32 participants, and has 290 classes for the active objects (annotation is not dense). We split the dataset into 217 train, and 55 val videos, and evaluate VOC mAP across annotated frames at three thresholds. Instance track ids are not provided in the dataset, so we disable our graph loss. We see a **+2.7 mAP** improvement at $IoU_{0.5}$ due to the long-term aggregation, as shown in Table 6.

5. Summary

We introduced a novel method of relation reasoning between object proposals in different frames that allows long-term feature support in video object detection. The resulting detectors better exploit longer-term temporal dependencies than previous work, at lower computational cost, only a small addition on top of per-frame detection alone. The system is evaluated in a "causal" setting with high accuracy. Our method for long-term relation reasoning between frames uncovers techniques that may be extended beyond video detection.

Acknowledgements: Nuro, UNC, and NSF grants 1452851, 1526367, and 1533771.

References

[1] Gedas Bertasius, Lorenzo Torresani, and Jianbo Shi. Object detection in video with spatiotemporal sampling networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 331–346, 2018.

[2] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. Scaling egocentric vision: The epic-kitchens dataset. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 720–736, 2018.

[3] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Detect to track and track to detect. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3038–3046, 2017.

[4] Wei Han, Pooya Khorrami, Tom Le Paine, Prajit Ramachandran, Mohammad Babaeizadeh, Honghui Shi, Jianan Li, Shuicheng Yan, and Thomas S Huang. Seq-nms for video object detection. *arXiv preprint arXiv:1602.08465*, 2016.

[5] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[7] Congrui Hetang, Hongwei Qin, Shaohui Liu, and Junjie Yan. Impression network for video object detection. *arXiv preprint arXiv:1712.05896*, 2017.

[8] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[9] Han Hu, Jiayuan Gu, Zheng Zhang, Jifeng Dai, and Yichen Wei. Relation networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3588–3597, 2018.

[10] Kai Kang, Hongsheng Li, Junjie Yan, Xingyu Zeng, Bin Yang, Tong Xiao, Cong Zhang, Zhe Wang, Ruohui Wang, Xiaogang Wang, et al. T-cnn: Tubelets with convolutional neural networks for object detection from videos. *IEEE Transactions on Circuits and Systems for Video Technology*, 28(10):2896–2907, 2018.

[11] K Kang H Li, T Xiao, W Ouyang, J Yan, X Liu, and X Wang. Object detection in videos with tubelet proposal networks. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognition, Hawaii, the US*, pages 727–735, 2017.

[12] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2117–2125, 2017.

[13] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[14] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[15] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[16] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[17] Adam Santoro, David Raposo, David G Barrett, Mateusz Malinowski, Razvan Pascanu, Peter Battaglia, and Timothy Lillicrap. A simple neural network module for relational reasoning. In *Advances in neural information processing systems*, pages 4967–4976, 2017.

[18] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

[19] Shiyao Wang, Yucong Zhou, Junjie Yan, and Zhidong Deng. Fully motion-aware network for video object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 542–557, 2018.

[20] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7794–7803, 2018.

[21] Xiaolong Wang and Abhinav Gupta. Videos as space-time region graphs. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 399–417, 2018.

[22] Chao-Yuan Wu, Christoph Feichtenhofer, Haoqi Fan, Kaiming He, Philipp Krahenbuhl, and Ross Girshick. Long-term feature banks for detailed video understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 284–293, 2019.

[23] Fanyi Xiao and Yong Jae Lee. Video object detection with an aligned spatial-temporal memory. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 485–501, 2018.

[24] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1492–1500, 2017.

[25] Xizhou Zhu, Jifeng Dai, Lu Yuan, and Yichen Wei. Towards high performance video object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7210–7218, 2018.

[26] Xizhou Zhu, Yujie Wang, Jifeng Dai, Lu Yuan, and Yichen Wei. Flow-guided feature aggregation for video object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 408–417, 2017.