Recurrent Multi-frame Single Shot Detector for Video Object Detection

Alexander Broad[1]
alex.broad@u.northwestern.edu
Michael Jones[2]
mjones@merl.com
Teng-Yok Lee[2]
tlee@merl.com

[1] Department of Electrical Engineering and Computer Science
Northwestern University
Evanston, IL USA

[2] Mitsubishi Electric Research Labs
201 Broadway, 8th floor
Cambridge, MA USA

Abstract

Deep convolutional neural networks (CNNs) have recently proven extremely capable of performing object detection in single-frame images. In this work, we extend a class of CNNs designed for static image object detection to multi-frame video object detection. Our Multi-frame Single Shot Detector (Mf-SSD) augments the Single Shot Detector (SSD) meta-architecture [16] to incorporate temporal information from video data. By adding a convolutional recurrent layer to an SSD architecture our model fuses features across multiple frames and takes advantage of the additional spatial and temporal information available in video data to improve the overall accuracy of the object detector. Our solution uses a fully convolutional network architecture to retain the impressive speed of single-frame SSDs. In particular, our Recurrent Mf-SSD (based on the SqueezeNet+ architecture) can perform object detection at 50 frames per second (FPS). This model improves upon a state-of-the-art SSD model by 2.7 percentage points in mean average precision (mAP) on the challenging KITTI dataset. Additional experiments demonstrate that our Mf-SSD can incorporate a wide range of underlying network architectures and that in each case, the multi-frame model consistently improves upon single-frame baselines. We further validate the efficacy of our RMf-SSD on the Caltech Pedestrians dataset and find similar improvements of 5% on the pre-defined Reasonable set.

1 Introduction

Object detection remains a core problem in the computer vision community. This is partially due to its inherent complexity as well as its potential for wide-ranging application. Recently, we have seen great progress in the accuracy and speed of object detection frameworks for static images [22, 27, 34], however, there is less work in the related field of object detection in the video domain. Video provides information not available in a single image such as additional appearance information (pixels) as well as motion and depth information. Each of these sources of information can potentially be exploited to improve detection accuracy. In this work we explore a novel method of incorporating the extra information from video data into real-time object detectors. In particular, we focus on augmenting a new class

(or meta-architecture [16]) of object detectors called Single Shot Detectors (e.g. SSD [22], YOLO [27], SqueezeDet [34], DSSD [9], YOLOv2 [26]). The SSD meta-architecture directly predicts object bounding boxes and class probabilities from an input image. This is in contrast to the multi-stage process standard in region proposal-plus-classification methods. Recent SSD architectures are fully convolutional which results in extremely fast run-time.

While SSDs can run at speeds equal to (or faster than) standard video frame rates, they do not incorporate the extra temporal and spatial information available from streaming video data. To remedy this issue, we feed a video sequence into the model and use a *convolutional recurrent layer* to fuse information from the multiple input frames. This layer (1) improves detection accuracy by taking advantage of the spatio-temporal information available from multiple sequential frames, and (2) maintains the fully convolutional nature of recent SSD networks, resulting in very fast execution times. Because this recurrent fusion layer can be incorporated into any single-frame SSD-based object detection framework, we call the resulting meta-architecture a Recurrent Multi-frame Single Shot Detector (Recurrent Mf-SSD). Importantly, this additional layer does not impact the ability to train the network in an end-to-end fashion.

2 Related Work

The vast majority of recent work in object detection has focused on single-frame images. Convolutional neural networks (CNN) have dominated recent progress. Two-stage CNN-based detectors such as R-CNN [12], Fast R-CNN [11], and Faster R-CNN [28] replaced scanning window detectors (Viola-Jones [33]) with a region proposal stage followed by a multi-class classifier. Single shot detectors, such as Overfeat [30], and more recently, YOLO [27], SSD [22], YOLOv2 [26], DSSD [9], R-FCN [21], and SqueezeDet [34], replaced the proposal-plus-classification paradigm with a regression formulation that directly estimates a set of bounding boxes and class labels. Both region proposal style detectors, and single shot detectors, have become fast (using modern GPUs) and reasonably accurate. For an in-depth comparison of modern convolutional object detectors see [16].

When developing object detection algorithms it is important to remember that, for many applications, the natural input is actually video (not single-frame images). However, standard practice is to treat each frame independently, and simply process the input one frame at a time. There has been a comparatively small amount of work on object detectors that explicitly take a sequence of multiple frames as input. Within this space, we refer to object detectors that integrate features from multiple frames as *feature-level* approaches [37]. This is in contrast to *box-level* techniques which operate on the sequence of bounding boxes produced by object detectors applied independently to multiple sequential frames. In the video object detection literature, there has been significantly more work on box-level methods. One example is the work of Han et al. [13] that replaces standard Non-Maximal Suppression (NMS) with one that incorporates bounding boxes from multiple frames. Tripathi et al. [32] describe another box-level technique that processes a sequence of object detector outputs using a recurrent network to improve object predictions. Kang et al. [18, 19] use the output from a single-frame detector to produce spatial-temporal "tubelets" that are further processed to generate improved box predictions. Similarly, Lu et al. [23] use feature maps from a single-frame detector within detected regions and pass these to a recurrent network that outputs new bounding boxes and class probabilities. All of these approaches are related to another class of box-level techniques known as tracking-by-detection [1, 3, 4, 15, 20].

The basic idea of these methods is to associate detections across the output of an object detector applied independently to sequential single-frame images to create tracks that can be used to remove false positives and restore missed detections. In contrast, we propose a method that directly fuses feature-level information from multiple frames to improve per-frame detections. Box-level approaches are orthogonal to this goal and can be applied in a post-processing phase to the output of any multi-frame detector.

In comparison to box-level methods, there have been relatively few feature-level approaches in video object detection. One such approach is the work of Zhu et al. [37] who use optical flow to warp feature maps computed from two input frames into correspondence. Modest improvements over a single-frame baseline are shown on the ImageNet VID dataset [29]. In a related paper [38] use intermediary *key frames* to avoid computing feature maps for every frame. This idea increases speed at the cost of accuracy. In the latest paper in this series, [39] combine these two orthogonal ideas with a spatially-adaptive feature computation to further improve results on ImageNet VID. Another approach by Feichtenhofer et al. [8] uses a deep network to combine detection and tracking to improve object detection in videos. Good results are reported on ImageNet VID.

3 Multi-frame Single Shot Detector

In this section we describe the meta-architecture of our Multi-frame Single Shot Detector (Mf-SSD). The Mf-SSD takes a video sequence as input and requires only a simple modification to the SSD meta-architecture to handle the change in input space. The SSD meta-architecture has two main components: a *feature extractor* network consisting of convolutional and pooling layers and a *detection head* consisting of convolutional layers. The feature extractor takes an image as input and outputs a set of feature maps. The detection head takes the feature maps and outputs a set of bounding boxes and object class probabilities that indicate the detected objects. The main modification we make to the base SSD architecture is to add a data fusion layer to the network directly after the feature extractor to integrate information from the sequence of input images. The data fusion technique can be an element-wise operation (e.g. `add` or `max`), a simple concatenation of feature maps, or a recurrent layer. The key factor is that the fusion layer must either be convolutional itself, or a basic operation that does not alter the fully convolutional nature of the full network. By enforcing this requirement on the network architecture, we can ensure that the network will remain extremely fast. The output of the data fusion layer is then fed into the standard SSD detection head which produces the final bounding boxes and classes for the most recent time-stamped image. Importantly, this change *does not require extra labeled training data as only the final time-stamped image needs labeled bounding boxes*. In this work, we particularly focus on an Mf-SSD implementation that uses a convolutional recurrent layer to fuse the information from multiple frames at the feature level.

3.1 Recurrent Multi-frame Single Shot Detector

A special class of our Multi-frame Single Shot Detector meta-architecture is the Recurrent Multi-frame Single Shot Detector (RMf-SSD) which fuses information from multiple sequential images through the use of a recurrent layer. A pictorial representation of an example RMf-SSD can be seen in Figure 1. A different feature extractor, such as one based on VGG [31], ResNet [14], YOLOv2 [26] or SSD [22], could be used instead. This architec-

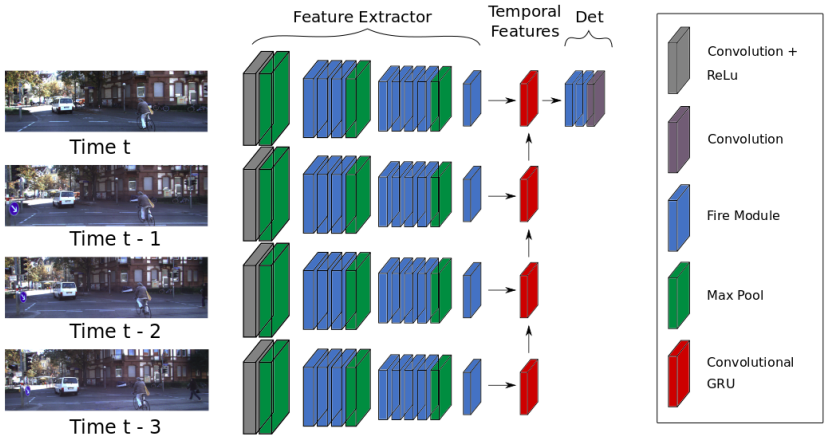


Figure 1: Recurrent Multi-frame Single Shot Detector (Mf-SSD) Meta-Architecture. The depicted RMf-SSD is based on the SqueezeDet+ architecture [34]. RMf-SSDs extend modern, single shot object detection frameworks to the video domain by merging information from sequential images with a convolutional recurrent fusion layer. The *fire module* refers to a specific combination of layers described in the original SqueezeNet paper [17].

ture takes a sequence of video frames as input and uses the same feature extractor on each sequential image to produce a set of feature maps. The feature maps from each image are then fed into the recurrent units along with the feature maps from the previous time step. The recurrent units output new feature maps which are then used to compute bounding boxes and class probabilities by more convolutional layers. This basic architecture allows the network to take advantage of information from previous frames to detect objects in the current frame.

Recurrent neural networks have demonstrated the ability to incorporate temporal information in many domains, and more recent advancements (such as LSTM and GRU units) have proven particularly successful. The Recurrent Mf-SSD uses convolutional recurrent units, instead of fully connected recurrent units, to maintain the fully convolutional structure of modern SSD architectures. We now describe convolutional recurrent units in more detail.

3.1.1 Convolutional Recurrent Units

Convolutional recurrent units combine the benefits of standard convolutional layers (i.e. sparsity of connection, suitability to spatial information) with the benefits of standard recurrent layers (i.e. learning temporal features). From a mathematical perspective, convolutional recurrent units simply replace the dot product operator in the standard fully connected recurrent unit definition with the convolution operator. In this work, we mainly focus on convolutional gated recurrent units (GRU) [2, 5]. We found LSTMs [35] and GRUs to yield similar accuracy but GRUs are faster to train. The updated GRU recurrent unit equations are:

$$z_t^l = \sigma(W_z * \boldsymbol{x_t} + U_z * \boldsymbol{h}_{t-1}), \qquad \tilde{h}_t^l = \tanh(W * \boldsymbol{x_t} + U * (\boldsymbol{r_t} \odot \boldsymbol{h}_{t-1})),$$
$$r_t^l = \sigma(W_r * \boldsymbol{x_t} + U_r * \boldsymbol{h}_{t-1}), \qquad h_t^l = (1 - z_t^l) h_{t-1}^l + z_t^l \tilde{h}_t^l$$

where $*$ denotes the convolution operator and $\odot$ denotes the Hadamard product. The variable z_t^l is an update gate, r_t^l is a reset gate, $\tilde{h}_t^l$ is a candidate activation, h_t^l is an output, and $\boldsymbol{x_t}$ is

the input (feature maps). The function σ is a logistic sigmoid function. W_z, W_r, W, U_z, U_r, and U are the convolution parameters that are learned. Ballas et al. [2] provide a more detailed treatment of convolutional GRU units, including information regarding the required computation and memory.

3.2 Implementation

We implement the described Recurrent Mf-SSD in Tensorflow on a machine running Ubuntu 16.04 with an nVidia Titan X (Pascal) GPU. We use SqueezeDet+[1] [34] as our baseline SSD model and, through experimentation, chose to use a Convolutional Gated Recurrent Unit[2] (GRU) as the convolutional recurrent layer. The SqueezeDet+ architecture was chosen as the baseline because it achieves state-of-the-art results on the KITTI dataset. Our implementation extended that of SqueezeDet+ so we can apply the same data preparation, training algorithms and parameters for fair comparison. A detailed description of the full network and further information on the fire module can be found in the supplementary material.

4 Experiments

We begin by evaluating the efficacy of our approach on the KITTI detection dataset [10]. We first compare the efficacy of our described Recurrent Multi-frame Single Shot Detector with a state-of-the-art single-frame baseline [34] and numerous alternative Mf-SSDs based on different data fusion techniques. We then perform further experimental analysis to demonstrate that RMf-SSDs provide consistent improvement over single-frame baselines, independent of the choice feature extractor (e.g. VGG-16 and ResNet-50). Finally, to validate our RMf-SSD in a different setting, we evaluate our RMf-SSD on the Caltech Pedestrian dataset.

4.0.1 KITTI

The KITTI [10] autonomous driving dataset consists of 6 hours of driving data under various weather conditions and traffic scenarios. Images were captured at 10 Hz with resolution 1242x375. The 2D object detection dataset consists of 7,481 training images and 7,518 testing images. Each image in this dataset is labeled with bounding boxes and class labels for three classes: cars, pedestrians and cyclists. KITTI is the *only* widely circulated video object detection dataset in which the *majority of images contain multiple objects and multiple classes*. Additionally, KITTI remains a challenging object detection dataset as it contains many small objects with varying amounts of occlusion, saturation, shadows and truncation. While the standard form of this dataset is best suited to single-frame object detection, KITTI does provide three prior frames of unlabeled data for each training image which we use to define the multi-frame video input. For our experiments on the KITTI detection dataset, we define train/val splits by randomly splitting the provided training data in half. Before doing so, we remove any images for which the prior three frames are not provided ($\sim$30 frames), as they are not suitable for video object detection. We evaluate each model on the validation set and use the standard evaluation metrics and provided evaluation code during analysis.

[1] https://github.com/BichenWuUCB/squeezeDet
[2] https://github.com/carlthome/tensorflow-convlstm-cell

4.1 Multi-Frame Fusion Techniques

To evaluate the efficacy of our described Recurrent Multi-frame Single Shot Detector, we compare and contrast our approach with a single-frame SSD baseline [34] that performs at the state-of-the-art on the KITTI dataset. Additionally, because it is unclear *a priori* which data fusion technique is the most appropriate when merging information from multiple frames [24, 25], we evaluate various alternative Mf-SSDs based on the aforementioned fusion techniques. In this work, we consider non-learnable techniques such as element wise `add` and `max` operations; learnable techniques such as early and late stage feature map concatenation; and lastly, we compare with a Recurrent Mf-SSD. The results of these experiments can be found in Table 1. Each of these experiments used 4 total frames to detect objects in the last frame.

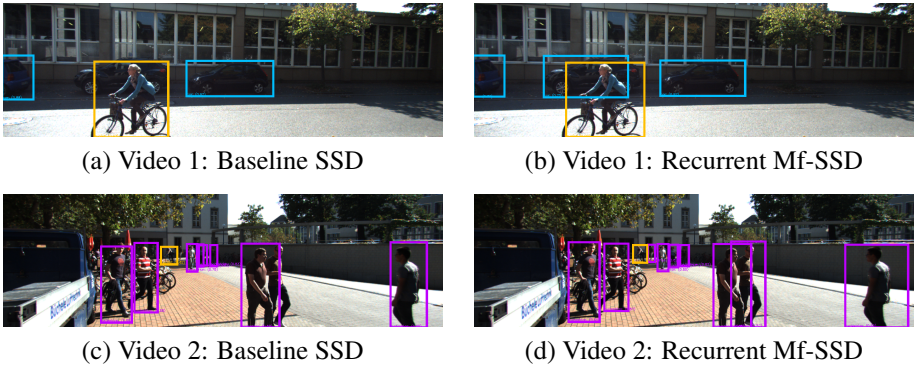
	Car			Pedestrian			Cyclist			
Method	E	M	H	E	M	H	E	M	H	mAP
Single-frame	0.933	0.885	0.798	0.858	0.775	0.741	0.872	0.845	0.789	0.833
Late Add	0.766	0.723	0.700	0.604	0.553	0.529	0.559	0.588	0.578	0.622
Late Max	0.798	0.780	0.707	0.597	0.547	0.537	0.529	0.560	0.534	0.621
Early Concat	0.896	0.881	0.796	0.850	0.772	0.743	0.866	0.840	0.793	0.826
Late Concat	0.920	0.886	0.799	0.867	0.782	0.751	**0.896**	**0.884**	0.851	0.849
Late Concat+Conv	0.906	0.888	0.851	**0.875**	**0.788**	0.757	0.890	**0.884**	0.848	0.854
Conv GRU	**0.946**	**0.891**	**0.870**	0.868	0.782	**0.758**	0.888	0.881	**0.853**	**0.860**

Table 1: KITTI Detection results. The top row represents the results of a single-frame SSD and each subsequent row represents the results of a multi-frame fusion technique. The input to each multi-frame network is the current frame and the three prior frames provided by KITTI for a total of four images. The only difference between each multi-frame network is the fusion technique. Our Recurrent Mf-SSD (last row), achieves highest mAP.

First, we evaluate element-wise `add` and `max` fusion operations. Here, we fuse the outputs of the feature extractor (see Figure 1) from each frame using element-wise operations on the 3D tensors. Both element-wise operations show a serious degradation in the overall accuracy of the model (Table 1, Lines 2&3). A simple explanation for these results is that element-wise operations do not solve the *correspondence problem* [6]. When we perform these operations on features from multiple frames we do not account for the relative shifts in image space, meaning these methods can easily obscure (in the case of `add`) or even destroy (in the case of `max`) important information. We provide evidence of this by using optical flow to solve the correspondence problem in the supplementary material.

The second set of methods we explore include the simple concatenation of features from different streams. Here, instead of performing element-wise operations on the feature maps, we stack the feature maps from each stream along the third dimension. In the results table, we relay the best performing early and late concatenation approaches (Table 1, Lines 4&5). The best early concatenation (i.e. before the end of the feature extractor) results come from fusing the streams after the second fire layer, and the best late concatenation (i.e. after the end of the feature extractor) results come from fusing the streams directly after the final layer in the feature extractor. Comparing these approaches to the single-frame baseline shows the late concatenation model does produce a significant increase in mean average precision, whereas early concatenation degrades the model's performance slightly. This is likely because it is

difficult to recover the pre-trained features earlier in the network.



(a) Video 1: Baseline SSD

(b) Video 1: Recurrent Mf-SSD

(c) Video 2: Baseline SSD

(d) Video 2: Recurrent Mf-SSD

Figure 2: Example detections. (a) The baseline single-frame model misses the car directly behind the cyclist, likely due to occlusion. (b) The Recurrent Mf-SSD model correctly detects all three cars, including the car occluded by the cyclist in the current frame. (c) The baseline single-frame model mistakenly detects only a single person in the foreground of the image, likely due to the proximity of the two pedestrians. (d) The Recurrent Mf-SSD model correctly detects both pedestrians in the foreground. The multi-frame approach utilizes information from prior images when the pedestrians are less occluded.

Finally, we discuss the results of our Recurrent Mf-SSD, which explicitly learns temporal relations between each stream by treating the resulting feature maps as a sequence which can be fed into a recurrent layer. This approach demonstrates the largest improvement in mean average precision of all of the methods tested (Table 1, Line 7). Using this model, we see an improvement of 2.7% mAP over the baseline. Figure 2 shows a few example differences between the detection capabilities of the single- and multi-frame approaches.

An important distinction between the recurrent approach and the other fusion techniques is that the recurrent layer inherently adds additional parameters to the network. For this reason we re-examine the best performing non-recurrent technique (Late Concat) with an additional convolutional layer defined by the same number of feature maps as the recurrent layer. This does indeed increase the mAP (Table 1, Line 6), however, we make two notes. First, by adding another convolutional layer to the network, we simply add depth to the detection sub-network. This is distinct from the use of additional parameters in the Recurrent Mf-SSD, which learn temporal correlations among feature maps extracted from each frame. Second, by using a stateful recurrent layer that processes frames sequentially we can simply feed video frames into the network as they are captured. This allows the Recurrent Mf-SSD to run at 50 FPS which is almost identical to the 55 FPS of the single-frame SqueezeDet+ network. In comparison, the "Late Concat+Conv" Mf-SSD network that uses 4 frames runs at 14 FPS because it computes feature maps for each frame. This could be sped up using a circular buffer, however, this increases the complexity of the model, increases the memory footprint, and does not scale to an arbitrary number of prior frames like recurrent layers do.

4.2 RMf-SSD using Various Feature Extractor Architectures

The second set of experiments we run validates the notion that the RMf-SSD meta-architecture can be integrated with *any* baseline architecture as a feature extractor. Specifically, we

supplement the results found in Table 1 with a comparison of single-frame SSDs to their multi-frame counterpart using SqueezeNet, VGG-16 and ResNet-50 as the feature extractor network instead of SqueezeNet+ (see Table 2).

		Car			Pedestrian			Cyclist			
Arch.	S/M	E	M	H	E	M	H	E	M	H	mAP
SqueezeNet	S	0.921	0.861	0.774	0.797	0.725	0.662	0.811	0.770	0.757	0.786
SqueezeNet	M	**0.936**	**0.884**	**0.798**	**0.856**	**0.769**	**0.741**	**0.860**	**0.831**	**0.794**	**0.830**
VGG-16	S	0.939	0.874	0.788	0.781	0.715	0.681	0.784	0.782	0.761	0.790
VGG-16	M	**0.942**	**0.886**	**0.798**	**0.815**	**0.731**	**0.695**	**0.810**	**0.801**	**0.792**	**0.808**
ResNet-50	S	0.926	0.852	0.751	0.772	0.718	0.672	0.781	0.776	0.745	0.777
ResNet-50	M	**0.946**	**0.870**	**0.782**	**0.778**	**0.730**	**0.679**	**0.785**	**0.779**	**0.765**	**0.791**

Table 2: KITTI Detection Results using alternative feature extractors. The second column represents whether the model takes a single-frame (S) or multiple-frames (M) as input. The RMf-SSD outperforms the baseline single frame network for all tested baseline architectures.

In Table 1, we see that the RMf-SSD based on SqueezeNet+ improves upon it's single-frame counterpart by 2.7% mAP. In Table 2, we see that the RMf-SSD based on SqueezeNet shows an improvement of 4.4%, the RMf-SSD based on VGG-16 shows improvement of 1.8% mAP, and the RMf-SSD based on ResNet-50 shows improvement of 1.4% mAP. By looking closer at Tables 1 and 2, we see that the each RMf-SSD model provides improvement over the respective single-frame model, both in overall mAP and in *every computed breakdown* by category and difficulty level. These results demonstrate a high level of consistency and show the ease of application of the Mf-SSD meta-architecture. All models are trained using the same procedure and hyper-parameters. Compared to SqueezeNet+, SqueezeNet has fewer channels per layer, which can achieve AlexNet-level accuracies [17]. To train the larger RMf-SSDs we reduce the input image size by half in each dimension. Additional information on the training procedure can be found in the supplementary material.

4.3 Analysis of Learned Temporal Features

(a) Frame 298. (b) Frame 299 with motion. (c) Frame 299 without motion.

Figure 3: Visualization of the feature map output from the recurrent layer in our RMf-SSD overlaid on input frames. The frames are from video 51 of the KITTI benchmark. The yellow highlights show that when motion information is present in the data (in the form of previous frames) the network responds to cars even if they are heavily occluded in the final frame. Without motion information, the occluded car is missed.

To analyze how temporal information is used in the recurrent layer, we designed an experiment to highlight how the activation levels of the recurrent layer change depending on

whether motion is present in the four input frames. This was done by giving the same trained RMf-SSD network either the usual four consecutive frames (t through $t-3$) or giving it the same frame t four times in a row. From the set of recurrent layer feature maps that differ depending on whether motion is present or not, we found several that can detect heavily occluded objects. Figure 3 shows one such feature map that detects cars from information in previous frames even when they are heavily occluded in the current frame. A visualization of the recurrent layer feature map computed from frames t to $t-3$ is overlaid on frame t (Figure 3.b) and $t-1$ (Figure 3.a). The yellow response shows that this feature map responds to the heavily occluded car behind the van. However, when we just give the same network frame t (with the occluded van) four times, the feature map has a low response (Figure 3.c).

4.4 Caltech Pedestrians

In addition to the experiments on the KITTI detection dataset, we include an evaluation of our Recurrent Mf-SSD on the Caltech Pedestrians dataset [7]. Caltech Pedestrians is also a multi-object dataset however it only contains a single class: pedestrians. The Caltech dataset is significantly larger than KITTI including 22,356 training images, however standard practice is to use a reduced set of training data (either 1/30th or 1/10th of the original data) and evaluate on 1/30th of the testing data which results in 4,204 testing images [36]. The Caltech Pedestrian images were captured at 30 Hz.

	SSD	RMf-SSD	RMf-SSD (2X)	Raw % Impr.
All	72%	70%	68%	4%
Reasonable	34%	32%	29%	5%

Table 3: Caltech Pedestrian results. Presented in Miss Rate (lower is better).

Table 3 shows results of both a single-frame SSD model (using the SqueezeNet+ architecture) and a Recurrent Mf-SSD on the full Caltech test set and the predefined *Reasonable* test set. Using this dataset, we can begin to explore the effect of frame-rate and the number of prior frames. In particular, when we use the three immediately prior frames, the RMf-SSD produces a small improvement relative to the baseline. However, if we use every other prior frame (i.e. t-2, t-4, t-6) we see a significantly larger improvement (RMf-SSD (2x)). Specifically, the RMf-SSD (2x) model outperforms the baseline by 5% on the *Reasonable* test set. Additional information regarding our analysis of the Caltech Pedestrian dataset can be found in the supplementary material.

5 Conclusions

In this work we have described a novel meta-architecture which we call a Recurrent Multi-frame Single Shot Detector (Recurrent Mf-SSD). Notably, it requires only a simple modification to the baseline Single Shot Detector meta-architecture to incorporate the additional spatio-temporal features available in video data. Because the Mf-SSD meta-architecture is fully convolutional, the runtime of the detection network remains very fast, an important aspect when working with video data. Furthermore, our approach is not slowed by the need to compute optical flow, unlike some previous approaches.

We have shown that our approach can incorporate additional temporal and spatial information from video datasets to improve upon state-of-the-art single-frame baselines on KITTI

while still running at 50 frames per second. Visualization of the feature maps output by the recurrent layer give some insights into how they take advantage of information across multiple frames. Furthermore, we tested numerous alternative methods of fusing information from multiple frames and found that the Recurrent Mf-SSD network is both the fastest and most accurate. Our empirical evaluation on the KITTI object detection framework demonstrates a significant improvement in overall accuracy over a state-of-the-art single-frame detector (2.7% mAP). Additionally, we find a similar improvement (5% on the Reasonable test set) on the related Caltech Pedestrians dataset.

References

[1] M Andriluca, S. Roth, and B. Schiele. People-tracking-by-detection and people-detection-by-tracking. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[2] Nicolas Ballas, Li Yao, Chris Pal, and Aaron Courville. Delving deeper into convolutional networks for learning video representations. *International Conference on Learning Representations*, 2016.

[3] J. Berclaz, F. Fleuret, and P. Fua. Robust people tracking with global trajectory optimization. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006.

[4] Michael Breitenstein, Fabian Reichlin, Bastian Leibe, Esther Koller-Meier, and Luc Van Gool. Online multi-person tracking-by-detection from a single, uncalibrated camera. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(9), 2010.

[5] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv:1412.3555*, 2014.

[6] Michael R Dawson. The how and why of what went where in apparent motion: Modeling solutions to the motion correspondence problem. *Psychological review*, 98(4), 1991.

[7] P. Dollár, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: A benchmark. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2009.

[8] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Detect to track and track to detect. In *ICCV*, 2017.

[9] Cheng-Yang Fu, Wei Liu, Ananth Ranga, Ambrish Tyagi, and Alexander C Berg. DSSD : Deconvolutional Single Shot Detector. *arXiv:1701.06659*, 2017.

[10] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[11] Ross Girshick. Fast r-cnn. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1440–1448, 2015.

[12] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 580–587, 2014.

[13] 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:1602.08465*, 2016.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[15] C. Huang, B. Wu, and R. Nevatia. Robust object tracking by hierarchical association of detection responses. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2008.

[16] Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, et al. Speed/accuracy trade-offs for modern convolutional object detectors. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[17] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and< 0.5 mb model size. *arXiv:1602.07360*, 2016.

[18] Kai Kang, Wanli Ouyang, Hongsheng Li, and Xiaogang Wang. Object detection from video tubelets with convolutional neural networks. In *Computer Vision and Pattern Recognition*, 2016.

[19] Kai Kang, Hongsheng Li, Tong Xiao, Wanli Ouyang, Junjie Yan, Xihui Liu, and Xiaogang Wang. Object detection in videos with tubelet proposal networks. In *Computer Vision and Pattern Recognition*, 2017.

[20] B. Leibe, K. Schindler, N. Cornelis, and L. Van Gool. Coupled object detection and tracking from static cameras and moving objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(10), 2008.

[21] Yi Li, Kaiming He, Jian Sun, et al. R-fcn: Object detection via region-based fully convolutional networks. In *Advances in Neural Information Processing Systems*, pages 379–387, 2016.

[22] 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.

[23] Yongyi Lu, Cewu Lu, and Chi-Keung Tang. Online video object detection using association lstm. In *ICCV*, 2017.

[24] Antoine Miech, Ivan Laptev, and Josef Sivic. Learnable pooling with context gating for video classification. *CoRR*, abs/1706.06905, 2017.

[25] Eunbyung Park, Xufeng Han, Tamara L Berg, and Alexander C Berg. Combining multiple sources of knowledge in deep cnns for action recognition. In *Winter Conference on Applications of Computer Vision (WACV)*, pages 1–8. IEEE, 2016.

[26] Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[27] 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.

[28] 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.

[29] 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. doi: 10.1007/s11263-015-0816-y.

[30] Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Robert Fergus, and Yann Lecun. Overfeat: Integrated recognition, localization and detection using convolutional networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, April 2014.

[31] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *International Conference on Learning Representations*, 2014.

[32] Subarna Tripathi, Zachary C Lipton, Serge Belongie, and Truong Nguyen. Context matters: Refining object detection in video with recurrent neural networks. *arXiv:1607.04648*, 2016.

[33] Paul Viola and Michael Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, 2004.

[34] Bichen Wu, Forrest Iandola, Peter H. Jin, and Kurt Keutzer. SqueezeDet: Unified, Small, Low Power Fully Convolutional Neural Networks for Real-Time Object Detection for Autonomous Driving. In *CVPR Workshops*, 2017.

[35] SHI Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Advances in Neural Information Processing Systems*, pages 802–810, 2015.

[36] Shanshan Zhang, Rodrigo Benenson, Mohamed Omran, Jan Hosang, and Bernt Schiele. How far are we from solving pedestrian detection? In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1259–1267. IEEE, 2016.

[37] Xizhou Zhu, Yujie Wang, Jifeng Dai, Lu Yuan, and Yichen Wei. Flow-guided feature aggregation for video object detection. In *ICCV*. IEEE, 2017.

[38] Xizhou Zhu, Y. Xiong, Jifeng Dai, Lu Yuan, and Yichen Wei. Deep feature flow for video recognition. In *CVPR*. IEEE, 2017.

[39] Xizhou Zhu, Jifeng Dai, Lu Yuan, and Yichen Wei. Towards high performance video object detection. In *CVPR*. IEEE, 2018.