Recurrent Neural Networks for object detection

Bin Qassim Ahmad
Technical University of Munich
Department of Informatics
Munich, Germany
email address

Pettirsch Arnd
Technical University of Munich
Department of Informatics
Munich, Germany
a.pettirsch@outlook.de

***Abstract*—ToDo**

***Index Terms*—TBD.**

I. Introduction

A. Image and Video Object Detection in general

ToDo

B. Recurrent Neural Networks in general

1) Delimitation to non-recurrent Neural Networks: Non-recurrent Neural Networks process on single inputs. In the field of image classification this could be for example a single image.
Recurrent Neural Networks process on sequences of date. In the field of classification those sequences consist often multiple frames. Recurrent Neural Network's core concept to enable the sequence processing is parameter sharing across different parts of a model. Parameter sharing can be reached by cycles in the architecture. **[?]**

2) Common Types of Recurent Neural Networks: Most of the paper which are described in this work use two pretty common approaches of Recurrent Neural Networks. The first one are LSTMs, first mentioned in **[?]** in 1991. The key of LSTMs is the inner cell state. LSTMs consist of four layers. Those four layers are able to remove or add information to the inner cell state. **[?]**

The second type of Recurrent Neural Networks is Gated Recurrent Units (GRUs). The main difference to LSTMs is, that those GRUs consist out of a single gated unit which can simultaneously control the forgetting factor and decide to update the inner cell state. **[?]**

II. Feature-based Video Object Detection

A. Definition

First, we want to introduce feature-based Video Object Detection methods. As defined, for example, in **[?]** feature-based Video Object Detection methods fuse detectors which integrate features from multiple frames into their video detection. In most of our papers those detectors use recurrent units to integrate the visual features from different frames.

B. Recurrent Multi-frame Single Shot Detector for Video Object Detection

Broad, Jones and Lee have in **[?]** and **[?]** the idea to insert a fusion-layer into a classical Single Shot Detector (SSD) meta-architecture. Based on this idea they research in two main fields: On the one hand, they investigate different fusion methods and on the other hand they try several SSD meta-architectures. The final architecture is shown in **Fig. 1**

SSDs is generally consist of two main components. First, an so called Feature Extractor which consist of convolutional and pooling layers. As an input the feature extractor gets an image and it outputs feature-maps. The other component is an Detection head which consists of convolutional layers and creates bounding boxes and class probabilities out of the feature maps created by the feature extractor.

As mentioned above Broad, Jones and Lee have the idea to insert a fusion layer in between those two components. As fusion techniques they test simple element-wise operations (e.g.: add or max), simple concatenations of features maps and recurrent layers. Their experiments show that the recurrent layers perform best. Element-wise operations do not lead to an improvement in the mAP. And concatenations just add another depth to the network, which increases the mAP slightly.Only the Recurrent Units add new parameters, to learn temporal context, to the Network. In addition, they observe that recurrent units do not slow down the computational speed significantly. The state of the art SSD, which is used to test the different fusion techniques performs on 55 fps the recurrent one on 50 fps. As a recurrent unit they use GRU because they observ that the results were similar to the results when they use LSTMs but the GRUs perform faster.

In addition to the investigation on the type of fusion layer, Broad, Jones and Lee test different types of SSDs as a baseline for their architecture. They find out that for all baseline SSDs the mAP was higher in comparison to the non-recurrent models. The mAP increase by 2.7 to 5 percent on KITTI dataset. The best results is made with SequeezeNet+ as a baseline SSD network.

Unfortunately, there is only a little information on the training of the Recurrent Multi-frame Single Shot Detector. They use the SequeezeNet Training strategies and a pre-trained version of the baseline SSD. Finally, they use the SequezzeNet fine tuning strategy to train the whole network afterwards.

Figure 1

C. Mobile Video Object Detection with Temporally Aware Feature Maps

Liu and Zhu have the goal to design a video object detection architecture, which can run real-time on low-powered mobile and embedded devices. The key of their method is to combine convolutional layers with convolutional LSTMs. On this core idea they do some further research. They test the benefit from adding an LSTM into the baseline SSD, different types of Recurrent Layers (LSTM, GRU and bottleneck LSTMs), different dimensions of their bottleneck LSTMs and different LSTM placement strategies (Single LSTM placement and multiple LSTM placement)

In the beginning they simply add one LSTM to their baseline SSDs architecture - MobileNet. They observe that adding the LSTM improves the mAP in comparison to their baseline SSD architecture. Moreover, they investigate that the greatest improve is by adding the LSTM after the 13th convolutional Layer of the SSD.

Afterwards, they compare LSTMs, GRUs and Bottleneck LSTMs as different types of Recurrent units by placing them after the 13th convolutional layer. Bottleneck-LSTMs have been designed by Liu and Zhu to increase the efficiency of LSTMs. For that purpose they use slightly unusual the ReLU function as activation for the LSTM. Moreover they, compute a so called Bottleneck feature map with less input channel than the originally feature map and feed this map into the LSTM to reduce computational power. They come to the conclusion that Bottleneck-LSTMs are more effective the LSTMs and in the case of a convolutional kernel greater 1x1 even more effective than GRUs while attaining comparable perfomance.

In addition to the bottleneck LSTMs Liu and Zhu extend their network with multipliers. They use α_{base}, α_{ssd} and α_{lstm} as multiplier to scale the channel dimension of each layer. During their research they find out that the accuracy of the model remains near constant up to $\alpha_{ssd} = 0.25\alpha$ which means that the output of each LSTM is one-fourth the size of the input. For the other multipliers they use the values: $\alpha_{base} = \alpha$ and $\alpha_{ssd} = 0.5\alpha$.

As shown in **Figure** the final model uses LSTMs after all feature maps, because Liu and Zhu observe that there a slight performance improve by adding LSTMs after every feature map and nearly no change in computational cost.

On the training strategy and the loss function Liu and Zhu do not provide any information.

D. Feature Selective Small Object Detection via Knowledge-based recurrent attentive neural networks

ToDo

E. Looking fast and slow: memory-guided mobile video object detection

Todo

F. Delving Deeper into Convolutional Networks for Learning Video Represenations

ToDo

G. Detect to Track and track to detect

ToDo

III. Box-Level-based Video Object Detection

A. Definition

ToDo

B. Object Detection from Video Tubelets with Convolutional Neural Networks

ToDo

C. Optimizing Video Object Detection via Scale-Time Lattice

ToDo

D. Context Matters: Refining Object Detection in Video with Recurrent Neural Networks

Tripathi, Lipton, Belongie, Nguyen come up with a neural network architecture for video-based object detection. Their architecture consists two parts: A pseudo-labeler, which assigns labels to all video frames and a recurrent unit which refines those pseudo-labels by using the contextual information. Moreover, they describe a training strategy for their architecture and compare their approach to other models on the YouTube-Objects dataset.

The final architecture can be found in **Fig 2**. Tripathi, Lipton and Belongie first train the pseudo-labeler, which is an YOLO object detection network originally trained for 20-class PASCAL VOC on the YouTube-Video Dataset. As specified in YOLO [] they minimize the weighted squared detection loss and optimize classification and localization error simultaneously.

After training the pseudo-labeler they train the Recurrent Neural Network, which takes as an input the pseudo-labels and outputs improved prediction. The RNN consists of two GRU layers.

For training the whole network they use the following loss function to take both accuracy at the target frame and consistency of predictions across adjacent time steps into consideration:

$$loss = d_{loss} + \alpha \cdot s_{loss} + \beta \cdot c_{loss} + \gamma \cdot pc_{loss}$$

For the final output Tripathi, Lipton, Belongie and Nguygen use the object detection loss as described in YOLO []:

ToDo

The similarity loss considers the dissimilarity between the pseudo-labels and prediction at each frame t:

ToDo

The category loss takes wrong class probabilities into consideration:

-

And the prediction-consistency loss regularizes the model by encouraging smoothness predictions across the time-steps:

-

During the evaluation they find two possible areas of improvement for their approach. On the one hand the RNN is not able to recover from wrong predictions made by the pseudo-labeler after they have been fed into the RNN. On the other hand, they observe that their network is not robust to motion.

E. *Spatially Supervised Recurrent Convolutional Neural Networks for Visual Object Tracking*

In []

IV. Flow-based Object Detection

A. *Definition*

Another type of architectures for Video Object Detection defined, for example in [], are architectures which use Flow-Networks to consider the temporal context. The flow network estimates the optical flow which means it projects back the location in the current frame to an earlier frame.

B. *Deep Feature Flow for Video Recognition*

Ning, Zhang, Huang, He, Ren and Wang design in [] a combination of box-level and feature-level based Video detectors. They use the YOLO network to create high-level visual features and preliminary location inferences and feed both into a recurrent unit. The architecture, called ROLO, is shown in **figure**

They use three phases to train their model: The pre-training phase of the convolutional layers for feature learning, the traditional YOLO training phase for object proposal and the LSTM training phase for object tracking.

In the pre-training phase the convolutional layers, which create the feature maps are training on ImageNet data. Afterwards the YOLO-architecture is adopted as detection module. At least they add LSTMs for tracking. The LSTMs are fed with, the feature representations from the convolutional layers, the Bounding boxes from the detection module and the output states from the LSTM in the previous time-step. For training they use the Mean Squared Error (MSE):

$$L_{MSE} = \frac{1}{n} TODO$$

As an alternative, Ning, Zhang, Huang, He, Ren and Wang mention a Heatmap as input for the LSTM. Therefor the prediction and the visual features are concatenated before adding them to the LSTM. This is helpful to visualize the intermediate results.

Special to this architecture is, that LSTM does regression in two folds. There is a regression within one frame between the location inferences and the high-level features and there is also regression over the different frames of the sequence.

V. Comparison of different approaches

A. *General*

ToDo

B. *Conclusion Perfomance*

Todo

C. *Conclusion Prediction Quality*

Todo

VI. Outro

A. *Conclusion*

Todo

B. *Further work*

Todo

Acknowledgment

Todo

References

Please number citations consecutively within brackets [1]. The sentence punctuation follows the bracket [2]. Refer simply to the reference number, as in [3]—do not use "Ref. [3]" or "reference [3]" except at the beginning of a sentence: "Reference [3] was the first . . ."

Number footnotes separately in superscripts. Place the actual footnote at the bottom of the column in which it was cited. Do not put footnotes in the abstract or reference list. Use letters for table footnotes.

Unless there are six authors or more give all authors' names; do not use "et al.". Papers that have not been published, even if they have been submitted for publication, should be cited as "unpublished" [4]. Papers that have been accepted for publication should be cited as "in press" [5]. Capitalize only the first word in a paper title, except for proper nouns and element symbols.

For papers published in translation journals, please give the English citation first, followed by the original foreign-language citation [6].

References

[1] G. Eason, B. Noble, and I. N. Sneddon, “On certain integrals of Lipschitz-Hankel type involving products of Bessel functions,” Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955.

[2] J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[3] I. S. Jacobs and C. P. Bean, “Fine particles, thin films and exchange anisotropy,” in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.

[4] K. Elissa, “Title of paper if known,” unpublished.

[5] R. Nicole, “Title of paper with only first word capitalized,” J. Name Stand. Abbrev., in press.

[6] Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, “Electron spectroscopy studies on magneto-optical media and plastic substrate interface,” IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].

[7] M. Young, The Technical Writer’s Handbook. Mill Valley, CA: University Science, 1989.