24th International Conference on Artificial Neural Networks (ICANN), Hamburg, Germany, September 2014

# Structured Prediction for Object Detection in Deep Neural Networks

Hannes Schulz and Sven Behnke

Rheinische Friedrich-Wilhelms-Universität Bonn
Institut für Informatik VI, Friedrich-Ebert-Allee 144
{schulz,behnke}@ais.uni-bonn.de

**Abstract.** Deep convolutional neural networks are currently applied to computer vision tasks, especially object detection. Due to the large dimensionality of the output space, four dimensions per bounding box of an object, classification techniques do not apply easily. We propose to adapt a structured loss function for neural network training which directly maximizes overlap of the prediction with ground truth bounding boxes. We show how this structured loss can be implemented efficiently, and demonstrate bounding box prediction on two of the Pascal VOC 2007 classes.

**Keywords:** deep learning, neural networks, object detection

## 1 Introduction

After great success in image classification, neural network research has recently turned to detecting instances of object categories in images. Here, the task is to predict a bounding box $y \in \mathbb{R}^4$ for every object of a given class. Object detection differs from image classification in one main aspect—the solution space is huge. In fact, it contains any number of bounding boxes with position, size, and aspect ratio. Correctness is typically defined by measuring overlap with the ground truth, such that more than one correct solution exists. Under these conditions, simply scaling the multinomial logistic loss, which worked for classification of the 1000-class ImageNet dataset, is not an option. Instead, a number of competing approaches have been proposed.

In this paper, we shortly review recently published methods for object detection with deep neural networks and emphasize that they, successfully, optimize heuristic surrogate loss functions. These surrogate loss functions are related to the overlap criterion via an accompanying post-processing step, which is not part of the training algorithm.

We then suggest an alternative method based on structured prediction, which optimizes the overlap criterion directly. We show how to determine bounding boxes efficiently while training and how to translate them into gradients. We evaluate our proposed method on two selected classes of the Pascal VOC 2007 dataset.

## 2 Related Work

Deep neural networks are increasingly applied to computer vision tasks such as image classification [1] or object-class segmentation [2]. Recent advances leading to success in the ImageNet challenge stem from *dropout* to prevent overfitting [3], rectifying linear units for improved convergence, backpropagation through max-pooling [4] and GPU implementations for speed, all of which are also used in this work.

Neural networks trained for an easy task can be adapted to more complex, but related tasks [5, 6]. Consequently, neural net-based object detection methods start with networks trained for classification [7, 8], or learn classification at the same time as detection [9].

Girshick et al. [8] use a pipeline for detection with selective search region proposals [10], which are warped and classified by a deep neural network. The learned features are then used as input to a support vector machine and outputs are ranked. In contrast to our method, this practice relies heavily on the potentially slow pre-segmentation to find object candidates. Sermanet et al. [9] regress on bounding box coordinates and classification confidence from a low-resolution output map. The network is run on six scales and produces many bounding box candidates, which are then merged by a heuristic merging operation for prediction. Here, we directly optimize bounding box overlap and associate predicted bounding boxes and ground truth objects while learning. Erhan et al. [11] produce a fixed set of bounding boxes during training, which are associated to the ground truth bounding boxes to determine the gradient. This method is closest to ours. However, we do not explicitly regress on bounding box coordinates and instead keep the direct correspondence between image and bounding box by using output maps. Our method also does not output a fixed set of bounding box candidates per image, since we infer their number from the network output. Finally, Szegedy et al. [7] construct targets for low-resolution output-maps, which encode the bounding box and its object quadrants. A heuristic function combines the outputs to produce the final bounding boxes. This approach requires one network per class and five times as many output maps as classes, which poses a potential scaling problem. Also, it is not clear whether the network can or should spend effort on refining a regression, when the desired output is a bounding box. Our proposed loss vanishes when the correct bounding box output is produced.

## 3 Structured Prediction for Object Detection

We start with a deep convolutional neural network, without fully-connected layers. The number of maps in the output layer is equal to the number of detectable object-classes in the dataset. Our goal will be to produce values in the output map from which the bounding boxes in the image can be inferred. For inference, which is also part of our learning procedure, we make use of the fact that the

space of bounding boxes is *structured*, i.e. can be searched efficiently. In the following description, we draw heavily on Lampert et al. [12, 13], and adapt their formalism to neural networks.

We are given a training set of tuples $(x^i, Y^i)_{i=1,\dots,n} \subset \mathcal{X} \times \mathbb{P}(\mathcal{Y})$, where $x$ is an image, $Y$ is a set of bounding boxes, $\mathcal{Y}$ contains all possible bounding boxes in $x$, and $\mathbb{P}(\mathcal{Y})$ is the powerset of $\mathcal{Y}$. The prediction function $g(x) : \mathcal{X} \mapsto \mathbb{P}(\mathcal{Y})$ should minimize the empirical loss over the training set

$$\mathbb{E}_{x,Y}\left[\Delta(g(x), Y)\right], \tag{1}$$

where $g(x) =: \hat{\mathcal{Y}}$ is the inference function. The evaluation criterion for detection is typically given as a 50% threshold on the Jaccard index of the prediction and a ground truth object,

$$A(\bar{y}, y) = \begin{cases} 0 & \text{if } \frac{\text{area}(\bar{y} \cap y)}{\text{area}(\bar{y} \cup y)} > \frac{1}{2} \\ 1 & \text{otherwise.} \end{cases} \tag{2}$$

We also penalize the case where not all objects in an image have been identified, or more objects were returned than present in the ground truth $Y$. This is formalized in the set loss $\Delta : \mathbb{P}(\mathcal{Y}) \times \mathbb{P}(\mathcal{Y}) \mapsto \mathbb{R}$. Lampert [13] suggests to use the max loss,

$$\Delta(Y, \hat{Y}) = \max_{y \in Y \ominus \hat{Y}} \lambda(Y, y), \tag{3}$$

where $Y \ominus \hat{Y}$ is the symmetric set difference and

$$\lambda(Y, y) = \begin{cases} 1 & \text{if } y \in Y \\ \min_{\bar{y} \in Y} A(\bar{y}, y) & \text{else.} \end{cases} \tag{4}$$

Expanding the symmetric set difference and simplifying slightly, we get

$$\Delta(Y, \hat{Y}) = \max\left(\min\left(1, \left|Y \setminus \hat{Y}\right|\right), \max_{y \in \hat{Y} \setminus Y} \min_{\bar{y} \in Y} A(\bar{y}, y)\right). \tag{5}$$

The first term accounts for objects in the ground truth for which no corresponding detection $\hat{y} \in \hat{Y}$ exists. The second term penalizes predictions which are not corresponding to ground truth objects. One possible problem here is that we could find one object as many times as there are objects in the image, using slightly different $\hat{y}$ that all overlap with one $y$. To prevent multile detection, we require that elements of $\hat{Y}$ have a maximum Jaccard index of 0.2, which can be enforced by greedily rejecting non-matching bounding boxes in the sequential inference procedure.

Next, we define a compatibility function $f$ between a neural network output map $N(x)$ and the bounding boxes $y \in Y$ over pixels $i, j$. A bounding box is a

mask $y$ on $N(x)$, where the rectangular part corresponding to the bounding box was set to 1, and all other values to 0:

$$f(x, Y, \theta) = \sum_{y \in Y} \sum_{ij} N_{ij}(x, \theta) \cdot y_{ij}. \tag{6}$$

For learning, we would like to find parameters $\theta$ s.t.

$$g(x) = \arg \max_{\hat{Y} \in \mathbb{P}(\mathcal{Y})} f(x, \hat{Y}, \theta) \approx y. \tag{7}$$

For a given training tuple $(x^i, Y^i)$, we can bound the loss using a hinge loss upper bound, as in Taskar et al. [14], obtaining

$$\begin{aligned}
\Delta(g(x^i), Y^i) &= \Delta\left(\arg \max_{Y \in \mathbb{P}(\mathcal{Y})} f(x^i, Y, \theta), Y^i\right) && (8)\\
&\leq \max_{\hat{Y} \in \mathbb{P}(\mathcal{Y})} \Big[\Delta(\hat{Y}, Y^i) - f(x, Y^i, \theta) + f(x, \hat{Y}, \theta)\Big] && (9)\\
&= \max_{\hat{Y} \in \mathbb{P}(\mathcal{Y})} \underbrace{\Big[\Delta(\hat{Y}, Y^i) + f(x, \hat{Y}, \theta)\Big]}_{H(N(x^i), \hat{Y}, Y^i)} - f(x, Y^i, \theta). && (10)
\end{aligned}$$

The role of the loss term is to ensure that the bounding boxes selected tend to have a bad detection measure, thereby forcing the network in $f(\cdot, \cdot, \cdot)$ to increase its margin over them.

The maximization in Eq. (10) can be performed as described in Lampert et al. [12], using branch-and-bound on $\mathcal{Y}$. Branch-and-bound recursively splits $\mathcal{Y}$ into subsets, which are described by two rectangles; $o_\cup$ describes the maximum extents of all bounding boxes in the set, whereas $o_\cap$ describes the minimum extents. For a given tuple $(o_\cup, o_\cap)$, we construct an upper bound $\overline{\Delta H}$ on the change in $\Delta$ caused by adding $\hat{y} \in (o_\cup, o_\cap)$ to $\hat{Y}$,

$$\max_{\hat{y} \in (o_\cup, o_\cap) \subset \mathcal{Y}} \Big[H(N(x), \hat{Y} \cup \{\hat{y}\}, Y) - H(N(x), \hat{Y}, Y)\Big] \leq \overline{\Delta H}(o_\cup, o_\cap, N(x), \hat{Y}, Y) \tag{11}$$

$$\begin{aligned}
&= F^+(N(x), o_\cup) - F^-(N(x), o_\cap)\\
&\quad + \max\left(\min(1, \max_{\bar{y} \in (o_\cup, o_\cap)} |Y \setminus (\hat{Y} \cup \{\bar{y}\})|), \min_{\bar{y} \in Y} \bar{A}(\bar{y}, o_\cap, o_\cup)\right) && (12)
\end{aligned}$$

where

$$\bar{A}(y, o_\cap, o_\cup) = \begin{cases} 0 & \text{if } \frac{y \cap o_\cup}{y \cup o_\cap} > \frac{1}{2} \\ 1 & \text{else,} \end{cases} \tag{13}$$

$$F^+(N(x), o_\cup) = \sum_{ij} \max(0, N_{ij}(x)) o_{\cup ij}, \text{ and} \tag{14}$$

$$F^-(N(x), o_\cap) = \sum_{ij} \min(0, N_{ij}(x)) o_{\cap ij}. \tag{15}$$

**Table 1.** Network Architecture for learning two classes.

| Layer | Output Size | Filter Size | Channels | Groups | Stride | Padding | Pool Size | Pool Stride |
|---|---|---|---|---|---|---|---|---|
| Input | 224×224 | – | 3 | – | – | – | – | – |
| Conv1 | 108×108 | 7 | 96 | 1 | 2 | 0 | 3 | 2 |
| Conv2 | 49×49 | 5 | 256 | 6 | 1 | 0 | 3 | 2 |
| Conv3 | 24×24 | 3 | 512 | 8 | 1 | 1 | – | – |
| Conv4 | 24×24 | 3 | 512 | 16 | 1 | 1 | – | – |
| Conv5 | 24×24 | 3 | 64 | 16 | 1 | 1 | – | – |
| Conv6 | 22×22 | 3 | 2 | 1 | 1 | 0 | – | – |

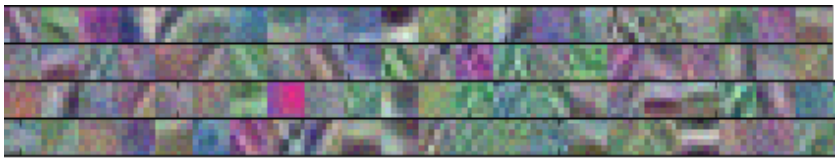

**Fig. 1.** Sample first-layer features of the classification network trained to discriminate between *cow* and *horse* images.

The rectangle sums $F^{\pm}(\cdot,\cdot)$ can be efficiently evaluated by pre-computing one integral image for each. A queue ensures that the search in $\mathcal{Y}$ is efficient.

Intuitively, learning proceeds by minimizing sums in the rectangles which are found by branch-and-bound on an output map $N(\cdot)$ and maximizing the sum of ground truth bounding boxes. Since bounding boxes overlapping with ground truth are given a disadvantage in Eq. (10), we focus on likely false detections (c.f. hard negatives) during optimization. Note that in contrast to Szegedy et al. [7], we do not need to specify on a per-pixel basis which value the output map should have, we only require the sums of regions to be higher or lower. The gradient consists of (differences of) rendered bounding boxes. For non-loss-augmented inference, the loss term is dropped from Eq. (10).

*Multi-Class Training with Weak Labelings.* In principle, all classes can be handled separately. Labels are typically weak, however, since not all objects are annotated. As commonly done, we only treat images $I$ as negative for class $c$ when no object of class $c$ is annotated in $I$. Additionally, if a ground truth bounding box $y$ is matched by $\hat{y}$ with a Jaccard index greater 0.5, we render the "negative" bounding box for $\hat{y}$ to refine the position found and eliminate the gradient in the overlap region. Since we intend to increase the margin of ground truth bounding boxes over others in $\mathcal{Y}$, we train only on images or parts of images which contain at least one annotated object.

## 4 Experiments

We present experiments on two difficult and easily confusable object classes, *cow* and *horse*. Our network architecture is shown in Table 1. It is roughly inspired

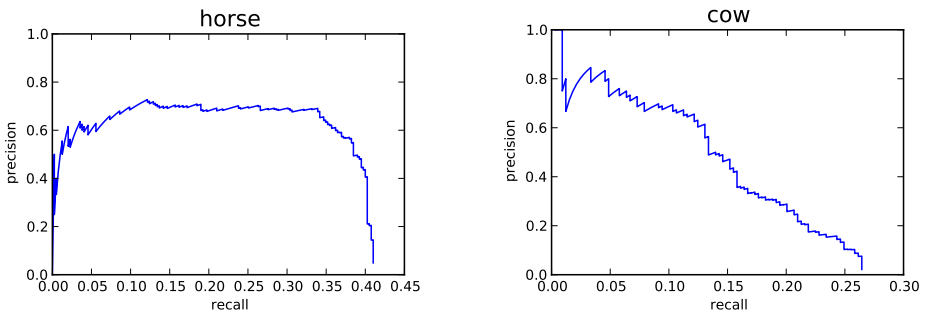


**Fig. 2.** Precision and recall curves for the two detected object classes *horse* and *cow* on the VOC 2007 test set, with average precision of 0.295 and 0.149, respectively.

by Krizhevsky et al. [1][1], but optimized for larger output maps. It also lacks the fully connected layers, which significantly reduces its capacity.

We pre-train the model with binary classification between the two classes we want to detect, using 8193 images from the Pascal10X database [15]. For this purpose, we add two fully connected layers with 1024 hidden neurons and MaxOut non-linearity [16], each. The network is trained using AdaGrad [17] to adapt the learning rates. The input images are scaled such that their shortest dimension is 256, then we extract a random crop of size 224×224. The images are flipped horizontally with a probability of 0.5. We perform this preprocessing in parallel on CPU, while the network runs on GPU. The network does not overfit on a held-out validation set (4466 images), but reduces the training error to zero. Figure 1 shows a subset of the learned first-layer features.

In a second step, we train the network to detect objects on the same split of Pascal10X, using the methods described in Section 3. Again, we use AdaGrad for learning rate adaptation, and a learning rate of 0.01. The bounding boxes in the dataset are scaled down by a factor of two before supplying them to the network to aid discrimination between neighboring objects. During loss-augmented inference, we find up to four bounding boxes with a Jaccard index of at most 0.2. The time required for loss-augmented inference amounts to approximately $1/7$ of the network evaluation time (forward and backward pass) when performed on CPU without any parallelization. As input we use (possibly flipped) images from three different scales in steps of $1/2$ octaves. The largest scale is chosen such that the original image corners are in the center of the receptive field of the corner output neurons. For this purpose, we add margin by mirroring parts of the image. Objects have to be at least two pixels wide when transformed to the output map, otherwise we do not use them as training examples on a given scale.

After convergence, we evaluate the network on the test set of the 2007 version of the Pascal Visual Object Classes Challenge [18] (402 images containing either a horse or a cow). Here, we use a sliding window on all three scales and combine the outputs within one scale by a $\max(\cdot,\cdot)$ operation. We scale all maps to the

[1] We would like to thank Alex Krizhevsky for making his code publicly available.

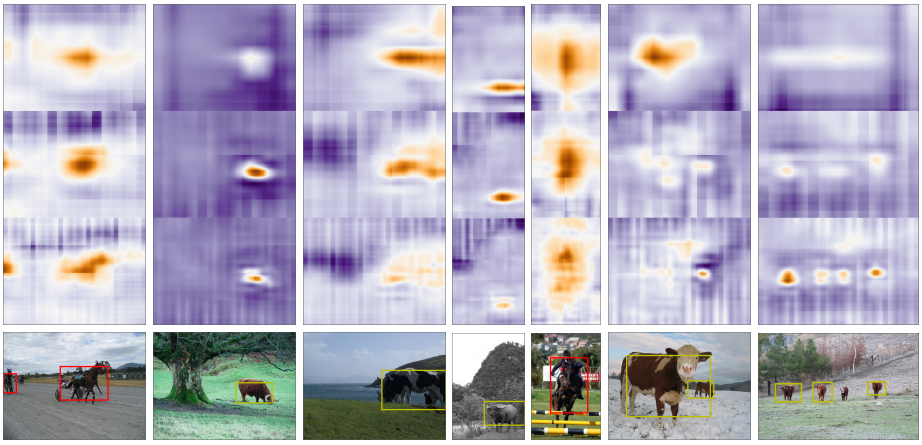

**Fig. 3.** Sample object detections. The figure shows the output map activations for the respective class. There are three scales, the lower two are processed with sliding window and merged using $\max(\cdot,\cdot)$. The bottom row shows the input image with detected bounding boxes.

same size. Following Szegedy et al. [7], we use $k$-means clustering on the bounding boxes in the training set and determine 10 reference bounding boxes, which we then scale by factors of $\{0.1, 0.2, \ldots, 0.9\}$. We slide all 90 bounding boxes over every scale and determine local maxima. Finally, we reduce the predicted set by removing bounding boxes which overlap with higher scoring ones by more than 20%. Figure 2 shows precision/recall curves for the two learned classes, and Figure 3 shows sample detections. The average precision for the *horse* class is comparable to Szegedy et al. [7] and Erhan et al. [11], however, both of these works trained on a complete VOC 2012 dataset.

## 5 Conclusion

Following success on ImageNet classification, there is much attention on adopting deep convolutional neural networks to perform more complex computer vision tasks, especially object detection. Multiple formulations have been proposed so far, to which we added our own. In contrast to previous work, we do not regress surrogate loss functions. Instead, we infer bounding boxes during training and minimize the overlap criterion directly, drawing heavily on previous work on structured prediction in the context of support vector machines.

We evaluate our model on two difficult classes of the VOC 2007 dataset, *cow* and *horse*, pretrained by classification, and show that with our formulation, a deep neural network can learn to localize instances of the two classes well. While results on two classes are not conclusive, we believe that this proof-of-concept shows that learning bounding boxes with structured prediction is feasible in deep neural networks. A more general pre-training on a larger dataset and wider architecture, as well as more network capacity and explicit handling of close-by objects is likely to improve the results significantly.

## References


[1] A. Krizhevsky, I. Sutskever, and G. Hinton. "Imagenet classification with deep convolutional neural networks". In: *Adv. in Neural Information Processing Systems*. 2012.

[2] H. Schulz and S. Behnke. "Learning object-class segmentation with convolutional neural networks". In: *Eur. Symp. on Art. Neural Networks*. 2012.

[3] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov. *Improving neural networks by preventing co-adaptation of feature detectors*. 2012. arXiv: 1207.0580.

[4] D. Scherer, A. Müller, and S. Behnke. "Evaluation of pooling operations in convolutional architectures for object recognition". In: *Artificial Neural Networks (ICANN), 20th Int. Conf. on*. 2010.

[5] G. E. Hinton and R. R. Salakhutdinov. "Reducing the dimensionality of data with neural networks". In: *Science* 313.5786 (2006).

[6] Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle, et al. "Greedy layer-wise training of deep networks". In: *Adv. in Neural Information Processing Systems* 19 (2007).

[7] C. Szegedy, A. Toshev, and D. Erhan. "Deep Neural Networks for Object Detection". In: *Adv. in Neural Information Processing Systems*. 2013.

[8] R. Girshick, J. Donahue, T. Darrell, and J. Malik. *Rich feature hierarchies for accurate object detection and semantic segmentation*. 2013. arXiv: 1311.2524.

[9] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. *OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks*. 2013. arXiv: 1312.6229.

[10] J. Uijlings, K. van de Sande, T Gevers, and A. Smeulders. "Selective search for object recognition". In: *Int. Journal of Computer Vision* 104.2 (2013).

[11] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov. *Scalable Object Detection using Deep Neural Networks*. 2013. arXiv: 1312.2249.

[12] C. H. Lampert, M. B. Blaschko, and T. Hofmann. "Efficient subwindow search: A branch and bound framework for object localization". In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 31.12 (2009).

[13] C. H. Lampert. "Maximum Margin Multi-Label Structured Prediction". In: *Adv. in Neural Information Processing Systems*. Vol. 11. 2011.

[14] B. Taskar, V. Chatalbashev, D. Koller, and C. Guestrin. "Learning structured prediction models: A large margin approach". In: *Int. Conf. on Machine Learning*. 2005.

[15] X. Zhu, C. Vondrick, D. Ramanan, and C. Fowlkes. "Do We Need More Training Data or Better Models for Object Detection?" In: *British Machine Vision Conference*. 2012.

[16] I. J. Goodfellow, D. Warde-Farley, M. Mirza, A. Courville, and Y. Bengio. "Maxout networks". In: *Int. Conf. on Machine Learning*. 2013.

[17] J. Duchi, E. Hazan, and Y. Singer. "Adaptive subgradient methods for online learning and stochastic optimization". In: *The Journal of Machine Learning Research* 12 (2011).

[18] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. "The pascal visual object classes (VOC) challenge". In: *Int. Journal of Computer Vision* 88.2 (2010).