OverFeat Implementation for Object Classification and Localization

Nathan Bargman[1] Kaitlyn Fichtner[1] Conrad Tulig[1]

Abstract

Object detection is an important task in computer vision that has been greatly improved with the use of deep neural network architectures. This project presents a single-scale implementation of OverFeat [1], a sliding window based approach to object classification, localization, and detection.

1. Introduction

Object classification, localization, and detection are critical tasks for many applications including autonomous driving, robotic grasping, surveillance, facial recognition, and many more. The state-of-the-art algorithms in this area of computer vision have been rapidly advancing with the development of deep convolutional neural networks and the computing power of GPUs [2].

OverFeat is one of the earliest successful approaches to deep learning based multi-scale object detection. It won the localization task in the ImageNet Large Scale Visual Recognition Challenge 2013 (ILSVRC2013) and achieved detection results in post-competition work that were state-of-the-art at the time [1]. While OverFeat was soon outperformed by architectures such as R-CNN [3] and YOLO [4], understanding OverFeat's sliding window approach can help explain how newer models advanced beyond sliding windows to achieve better results and efficiencies.

2. Related Work

Although OverFeat is now an outdated, we felt it is an important precursor to understanding differences and improvements other, more modern, models have made. Fast, Faster, and Mask R-CNN [8, 9, 10] all build off of the architecture R-CNN [3], that was developed around the same time as OverFeat. R-CNN utilizes a similar convolutional neural network structure to OverFeat; however, the classification and bounding box predictions are not made based off the same feature extraction network. Instead, R-CNN found that a more effective strategy is extracting region proposals first before passing each region of interest into the feature extraction and classification network to predict classifications for each region.

3. Proposed Method

Due to limited computational resources, we chose to use OverFeat's *fast* model at only a single scale. However, we still implement some features of OverFeat designed for multi-scale classification, so that we can learn the complexities of the multi-scale approach.

To pre-process the input data, each image is first resized so that the smallest dimension is 256 pixels. Next, five crops (4 corners and center) of size 231x231x3 and their horizontal flips are taken from each image to be used as the inputs to the neural network.

3.1 Feature Extraction

Both the classification and localization tasks utilize the same first five layers, shown in *Figure 1,* for feature extraction. These layers are similar to Krizhevsky et *al.*'s AlexNet architecture [7] and each of these layers use rectification ("relu") non-linearities and $\ell 2$ weight decay of $1x10^{-5}$ [1].

After the fifth layer's convolution, we use a custom max pooling function that creates four pooled feature maps by computing max pooling operations for each combination of Δx, Δy pixel offsets $\{0, 1\}$. We use two offsets instead of the three offsets $\{0, 1, 2\}$ described in OverFeat, because we are using a smaller input size than the multi-scale model for which the three offsets are intended. For the same reason, we use a 2x2 max pooling operation compared to the multi-scale model which uses 3x3 pooling at this step.

[1] Worcester Polytechnic Institute. *CS/DS 541 Deep Learning Final Project*, Worcester Polytechnic Institute.

Layer	1	2	3	4	5
Stage	conv + max	conv + max	conv	conv	conv + max
# channels	96	256	512	1024	1024
Filter size	11x11	5x5	3x3	3x3	3x3
Conv. stride	4x4	1x1	1x1	1x1	1x1
Pooling size	2x2	2x2	-	-	2x2
Pooling stride	2x2	2x2	-	-	2x2
Zero-Padding size	-	-	1x1x1x1	1x1x1x1	1x1x1x1
Spatial input size	231x231	24x24	12x12	12x12	12x12

Figure 1 - OverFeat feature extraction layers for fast model (figure from Sermanet et al. [1])

Next, a 5x5 sliding window is applied to each of the four pooled feature maps. These windows serve as inputs to the classification and localization models.

3.2 Classification

The classification network consists of three fully connected layers of 3072, 4096, and C (number of classes) nodes, respectively. The 3072 and 4096 node layers use relu activations and are each followed by a dropout layer with a rate of 0.5. The output layer uses a softmax activation function to produce the probabilities for each class. Each 5x5 window from the feature extractor is inputted to this classification network. As a result, each image will have multiple predictions for the different crops, pixel offset combinations, and sliding windows. We then take the element-wise maximum over all C-dimensional output vectors to get the final classification for the image. During training, the weights of the feature extractor and classifier are updated by stochastic gradient descent with categorical cross entropy loss.

3.3 Localization

Each 5x5 window from the feature extractor is also used as input to the localization regression network. The regression model consists of three fully connected layers of 4096, 1024, and 4 nodes, respectively. The output vector contains the four coordinates for the predicted bounding box (X_{min}, X_{max}, Y_{min}, Y_{max}). During training, the classification-trained weights are used for the feature extractor and the weights of the regression layers are updated by stochastic gradient descent using the $\ell 2$ loss between the predicted and ground truth bounding boxes.

The multiple bounding box predictions are combined using a greedy merge strategy, shown in *Figure 2*, which takes the average of all predicted bounding boxes with a *match score* below a certain threshold.

Bounding Box Merge Algorithm

```
b1 ← predictions[0]
for i in range(1, len(predictions)) do
  b2 ← predictions[i]
  match score = overlap between b1 & b2 + distance between b1 & b2
  if match score < threshold then
    b1 = (b1+b2)/2
return b1
```

Figure 2- Bounding Box Merge Algorithm

4. Experiments

4.1 Datasets

While OverFeat uses ImageNet, a dataset containing 1.2 million images across 1000 classes, we needed a much smaller dataset to compensate for our memory constraints. The images also needed to be at least 256x256 pixels or larger and have both class and bounding box labels. We tested and trained the classification network on *Imagenette* [5], a 10,000 image 10 class subset of ImageNet, and the *Cats and Dogs Breeds Oxford Dataset* [6], containing 7,400 images of 37 breeds of cats and dogs. When implementing localization, we could only use the *Cats and Dogs* dataset, since *Imagenette* did not have bounding box labels.

4.2 Feature Extraction

Although we initially implemented fine stride Δx, Δy pixel offsets {0, 1} for the purpose of understanding how this feature of the multi-scale model works, generating predictions for each 5x5 window within the pooled feature map from each offset used too much memory. As a result we decided to only use one offset ($\Delta x=0$, $\Delta y=0$) and only two 5x5 windows (from the top left and bottom left of the pooled feature map) for predicting. This essentially acts as the coarse stride approach with $\Delta=0$ described in OverFeat.

4.3 Classification

When initially constructing our classification model, we took the elementwise maximum of predictions across each offset and sliding window for an individual crop and then averaged the predictions between different crops of a given image. This is how OverFeat combines classification predictions; however, we found that averaging the predictions caused all elements in the output vector to be close to zero early in training, making it challenging for the model to learn. Instead, we took the maximum across all crops, offsets, and 5x5 windows. Our model was able to learn more easily after making this modification.

On the *Imagenette* dataset, we utilized the same learning rate scheduler as recommended in OverFeat. A learning rate decay of 0.5 was applied after 30, 50, 60, 70 and 80 epochs. However, we reduced the initial learning rate to $1x10^{-3}$ from the learning rate of $5x10^{-2}$ that is recommended in OverFeat. We also found that training on *Imagenette* data for 200 epochs with a batch size of 32 produced the best results.

In hyperparameter tuning on the *Cats and Dogs Breeds Oxford Dataset*, we reduced the learning rate decay rate to 0.25 which we applied after 30, 50, 70, and 90 epochs with an initial learning rate of $1x10^{-3}$. For this dataset, the best classification accuracies were achieved with 100 epochs and a batch size of 64.

4.3 Localization

The same learning rate decay used for classifying the *Cats and Dogs Breeds Oxford Dataset* was used when training the localization regression model. The best hyperparameters for training the regression network were 200 epochs and a batch size of 64. We also found that a threshold value of 53 produced good results in bounding box merging.

5. Results

5.1 Classification

The results from classifying ten classes in the *Imagenette* dataset are shown in *Figure 3*. On this dataset, our implementation reached an accuracy of 60.48%.

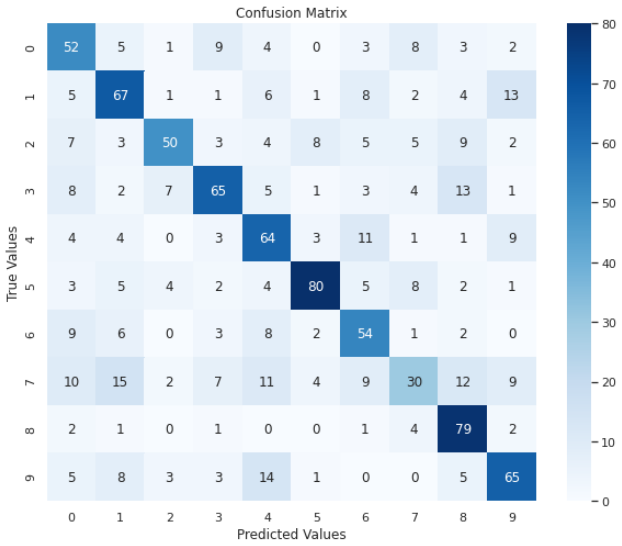

Figure 3 - Classification results on Imagenette with 60.48% test accuracy

On the *Cats and Dogs Breeds Oxford Dataset*, we reached an accuracy of 11.43% over 37 classes. This dataset was more difficult to classify because it contains more classes than *Imagenette* and it is more challenging to differentiate between different breeds of cats and dogs.

5.2 Localization

We used the *Cats and Dogs Breeds Oxford Dataset* to test our localization model. The dataset provided bounding box labels for locating the faces of each cat or dog. We achieved a lowest validation loss of 0.8221. However, we cannot report a test loss because the dataset did not have ground truth bounding box labels on the test images.

In many cases, the model was able to successfully predict where the dog and cat faces are located, as shown in *Figure 4*.

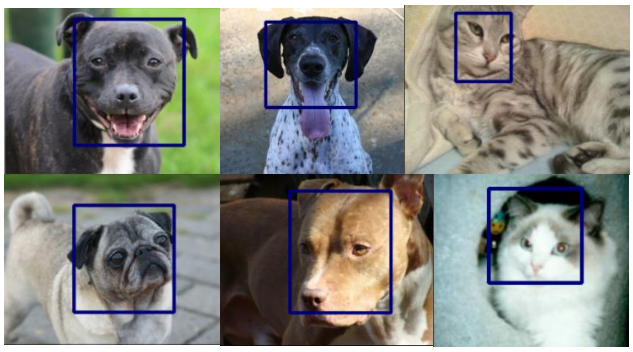

Figure 4 – Successful bounding box predictions

However, other predictions were slightly or completely off the correct location, as shown in *Figure 5*.

Figure 5 – Incorrect bounding box predictions

6. Discussion

We cannot directly compare our results to OverFeat's results on ImageNet because our data contains a different number classes and images. However, OverFeat was able to improve the accuracy of their single-scale model with a multi-scale approach and we believe that the multi-scale model would improve our accuracy as well.

The main limitation of this project is a lack of computational resources. We were not able to implement the multi-scale model for this reason. Additionally, most object detection datasets, such as ImageNet, are very large and it was challenging to find smaller subsets of these datasets. Consequently, we had very limited choices of what data to use and the amount of data we could utilize in training.

7. Conclusion

In future work, we would like to expand our classification and localization implementation to include object detection. We would also like to test more modern approaches to object detection, such as Faster R-CNN, Mask R-CNN or YOLO [4].

References

[1] Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., & LeCun, Y. (2013). OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks.

[2] Jiao, L., Zhang, F., Yang, S., & Li, L. (2019). A Survey of Deep Learning-based Object Detection. Retrieved from https://arxiv.org/pdf/1907.09408.pdf.

[3] Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. 2014 IEEE Conference on Computer Vision and Pattern Recognition. doi:10.1109/cvpr.2014.81

[4] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). doi:10.1109/cvpr.2016.91

[5] Howard, J., & Husain, H. (2019). Imagenette. Retrieved May 10, 2021, from https://github.com/fastai/imagenette.

[6] Avicenna, Dr. (2019). Cats and Dogs Breeds Classification Oxford Dataset. Retrieved May 10, 2021, from https://www.kaggle.com/zippyz/cats-and-dogs-breeds-classification-oxford-dataset.

[7] Krizhevsky, A., Sutskever, I., & Hinton, G. (2017). ImageNet classification with deep convolutional neural networks. Communications of the ACM, 60(6), 84–90. https://doi.org/10.1145/3065386

[8] Girshick, R. (2015). Fast R-CNN. 2015 IEEE International Conference on Computer Vision (ICCV). doi:10.1109/iccv.2015.169

[9] Ren, S., He, K., Girshick, R., & Sun, J. (2017). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(6), 1137-1149. doi:10.1109/tpami.2016.2577031

[10] He, K., Gkioxari, G., Dollar, P., & Girshick, R. (2017). Mask R-CNN. 2017 IEEE International Conference on Computer Vision (ICCV). doi:10.1109/iccv.2017.322