VGG Adaptation for Small Scale Scene Recongition

Elliot Bartel
Massachusetts Institute of Technology
kwbartel@mit.edu

Hansa Srinivasan
Massachusetts Institute of Technology
hansa@mit.edu

Abstract

In this work we adapt one of the most successful nets at classifying the ImageNet database to be suitable for scene classification in the Mini Places Challenge. Our main contribution is the tactical reduction of the VGG-16 ConvNet such that it does not overfit when trained on the Mini Places database, which is substantially smaller than ImageNet. By employing multi-cropping techniques and combining the classifications of separately trained models at test time, we achieve a a 45.3% top-1 error and a 18.43% top-5 error. These results secured our team's first place position in the Mini Places 2015 Challenge.

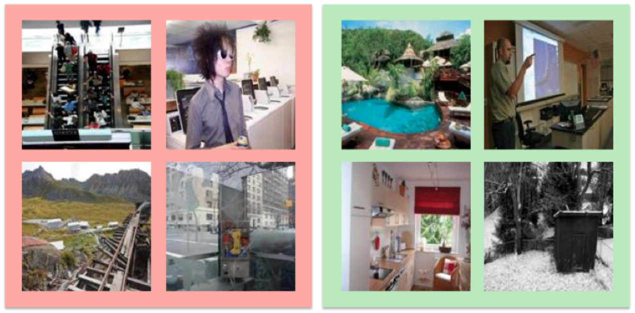
Figure 1. Example top-1 classification labels generated using our models. Images on the left were respectively misclassified as clothing store, office, construction site, and mountain (clockwise, from top left.) Images on the right were correctly classified.

1. Introduction

Deep learning through Convolutional Neural Nets (CNNs) is the state of the art approach to image classification and object recognition. The recent accessibility of large-scale image databases such as ImageNet and the increasing utilization of CNNs in computer vision has led to enormous strides in the area of object recognition. The top two competitors in the Imagenet Large Scale Visual Recognition Challenge (ILSVRC) 2014, VGG and GoogLeNet, achieved 7.4% and 6.8% top-1 error rates respectively [1]. However, scene recognition has not seen as dramatic a leap in classification accuracy, partially due to the lack of large scale image databases focused exclusively on scenes. The original Places database sought to rectify this absence by introducing the then-largest database of scene classified images, with over 8 million labeled images from 476 categories [2]. Places has now been succeeded by Places2, which contains over 10 million images from over 400 scene categories [3]. The Mini Places database has only 100 scene categories and is a strict subset of images from Places2.

2. Related Work

ConvNets were first successfully applied to object recognition in the form of AlexNet, an eight layer (five convolutional, three fully connected) net which demonstrated the capacity and efficiency of ConvNets and classified ImageNet images with a 15.3% top-5 error rate [4]. AlexNet set the standard for image recognition CNN architecture, including: employing frequent max-pooling layers after convolutional layers, a series of fully connected layers at the end of the net, using dropout to prevent overfitting in the fully connected layers, and using the Rectified Linear Unit (ReLU) model to reduce training times [4]. Another key strategy AlexNet introduced is using data augmentation while training to effectively increase the amount of training data.

Since then, many net structures have surpassed AlexNet in object recognition performance. The current top two structures are VGG and GoogLeNet [1]. These ConvNets are currently the two structures which can most accurately classify images from the ImageNet database. These structures are also the most successful systems when applied to large scale scene classification, though their accuracy rates are substantially lower. VGG-16 only achieves a top-1 error of 57% [3]. In addition, when models intended for large-scale classification are trained on much a smaller database such as the Mini Places set, they overfit to the training data and perform poorly on validation and test sets.

VGG differs from AlexNet in several key ways. Its kernel size is small and constant in each convolutional layer,

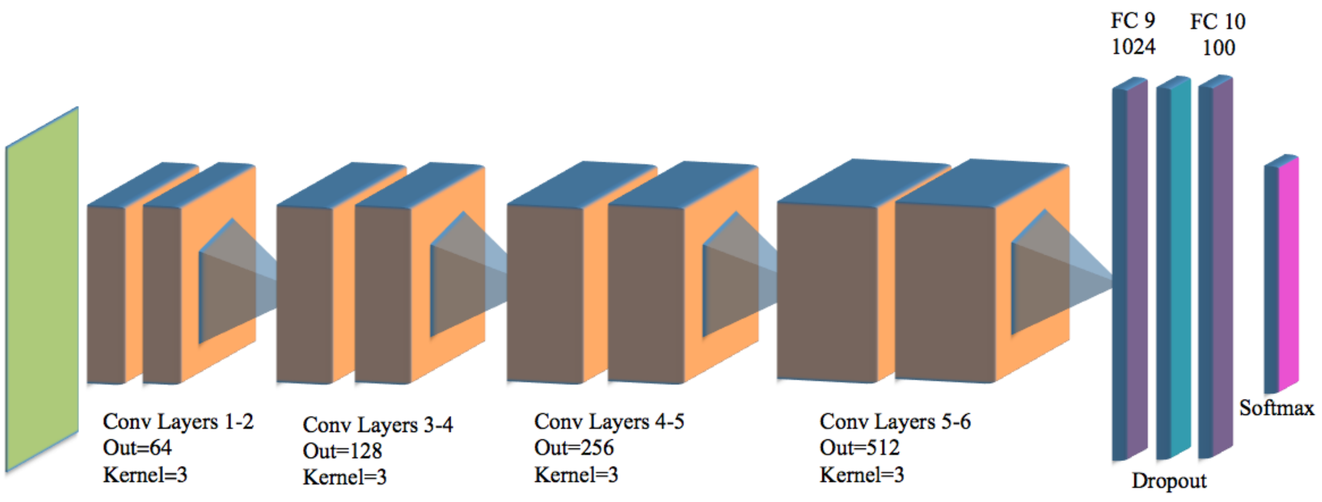

Figure 2. The WW4-VGG architecture.

while AlexNet's kernel size decreases from eleven to three over the course of several layers. Having VGG's convolutional layers follow one another directly in pairs and triples without max pooling allows for an effectively larger kernel size to be achieved with fewer parameters and more rectification. Fewer parameters helps avoid overfitting, and the additional rectification makes the net more discriminative [5]. An additional difference is that VGG-16 is sixteen layers long, twice as long as AlexNet. This allows VGG to make more high level connections based on abstractions farther removed from the original image, which is important in scene recognition. These key differences give VGG its advantage over AlexNet.

3. Neural Net Architectures

Our goal is to employ the small kernel size and relative depth which led VGG-16 to its success but tailor the structure to the size of the Mini Places database by reducing the depth and number of parameters of the model.

3.1. VGG-16 Adaptation

Refnet did not appear to overfit while training, so we aimed to adapt VGG-16 to have a similar number of parameters as Refnet.

To achieve a lower number of parameters, we made several changes to the VGG-16 architecture. We removed convolutional layers eleven through thirteen (size 512) and their max pooling, as well as the tenth (size 512) and seventh (size 256) convolutional layers. Finally, we removed a fully connected layer and modified the output size of the remaining two layers to be 1024 and 100, respectively. ReLU is used in each convolutional layer, and we borrow convolutional and maxpooling parameters directly from VGG-16. Thus we used 3×3 kernel size with a pad and stride of 1 for convolution, and a kernel of 2 with stride 2 for maxpooling. We also maintain VGG-16's dropout probability of

VGG Adaptations		
VGG-16	Deep WW4-VGG	Shallow WW4-VGG
16 weight layer	10 weight layers	8 weight layers
input (100 x 100 RGB image)		
conv3-64	conv3-64	conv3-64
conv3-64	conv3-64	conv3-64
maxpool (2 x 2, stride: 2)		
conv3-128	conv3-128	conv3-128
conv3-128	conv3-128	conv3-128
maxpool (2 x 2, stride: 2)		
conv3-256	conv3-256	conv3-256
conv3-256	conv3-256	conv3-256
conv3-256		
maxpool (2 x 2, stride: 2)		
conv3-512	conv3-512	conv3-512
conv3-512	conv3-512	conv3-512
conv3-512		
maxpool (2 x 2, stride: 2)		
conv3-512	conv3-512	
conv3-512	conv3-512	
conv3-512		
maxpool (2 x 2, stride: 2)		
FC-1024		
FC-100		
softmax		

Table 1. Model architectures. Side-by-side comparison of the original VGG-16 architecture [5] and our adaptations.

0.5 [5]. Table 1 demonstrates a side-by-side comparison of the original VGG-16 and our modified WW4-VGG model. The result of these modifications is a net with 3.72 million parameters, on the order of Refnet's roughly 2.69 million.

We hypothesized that removing layers from back would not harm performance because those layers are where high level associations are made, of which our net needs fewer due to the relatively small number of categories in Mini Places. We also removed the first fully connected layer because they contribute many learning parameters. We also altered the size of the last fully connected layer to be the number of categories in Mini Places.

We refer to this model as either WW4-VGG or Shallow WW4-VGG in Tables 1 through 3.

3.2. Deeper VGG-16 Adaptation

VGG-16 was built upon an 11 layer model with a high error rate. The additional layers between VGG-11 and VGG-16 led to a 4% increase in top-1 accuracy [5]. We aimed to improve our shallower VGG adaptation by adding layers in a similar manner to create our Deep WW4-VGG model. We added two 3×3 512 layers and a maxpool before the first fully connected layer, as shown in Table 1. These changes roughly doubled the number of learning parameters in the model, as demonstrated in Table 2.

4. Classification Framework

4.1. Training WW4-VGG

We trained the Shallow WW4-VGG model with learning parameters used to train the original VGG. The ConvNet was trained using mini-batch stochastic gradient descent with a momentum of 0.9. Our batch size differs from the original VGG training setup in that we used a batch size of 64 due to computational restrictions. We trained two versions of Shallow WW4-VGG from scratch, the first with a step size of 20,000. This setup trained poorly and its loss function was erratic and did not steadily decrease. Thus we decided to train the latter with a step size of 50,000, as used in Zhou *et al.* [6] This setup's loss function decreased steadily and was trained for 130,000 iterations before essentially converging to its final state.

It is possible that the differing step size allowed the second net to train more effectively than the first, but it is equally likely that the random weight initialization gave the second net a unique advantage. Random weight initialization has the potential to seriously impair or improve a net's ability to train due to the "instability of gradient in deep nets" [5]. We also used data augmentation in the form of randomly cropping the 128×128 original images down to 100×100, and performing random horizontal flips on the input training data.

4.2. Training Deep WW4-VGG

We trained WW4-VGG Deep by fine-tuning off the WW4-VGG Shallow model to decrease the necessary training time and ensure that the deeper net would not be affected by poor weight initialization. Thus we initialized the weights of WW4-VGG's first eight convolutional layers to match the WW4-VGG Shallow model, and initialized all subsequent layers randomly. We used the same learning parameters as the shallow model described in the previous section, and ran fine-tuning for 50,000 iterations. We decided to separately fine-tune an additional deep model with identical learning parameters with the exception of the initial learning rate, which we decreased to 0.001 to help avoid overfitting. This version ran for roughly 36,000 iterations. We refer to the separately trained WW4-VGG Deep structure with a smaller learning rate as Deep WW4-VGG-B, and the original as Deep WW4-VGG-A.

4.3. Testing

Once the ConvNets have been trained as described, we classify an image as follows. First, we augment the 128×128 image by scaling the original down to 100×100, taking five 100×100 crops of the original 128×128 (one crop for each corner, one crop centered), and additionally horizontally flipping all five crops. This results in eleven 100×100 augmented versions of the original. Second, each augmented image is passed as input through a net. We then average the eleven resulting scene classification probabilities of each cropped image, resulting in an updated scene probability estimate. Third, we pass the same set of eleven augmented images through a second, distinct net to be classified. Finally, we average the outputs of the different nets, take a soft max, and return the final set of classifications.

4.4. Implementation

We used the Caffe toolbox (branched September 2015) [7] for training the nets and created our own testing-classification framework using the Python Caffe wrapper library, which allowed us to augment the input images and combine outputs from different nets to receive our final classifications. Our final results are saved as a text file in the Mini Places specified format, which we then tested using the provided Mini Places test framework. Our Caffe models can be found at `https://github.com/hansa0/WW4_mini_places`.

5. Experimental Results

Here we show the classification results by applying the described ConvNet architectures to the the Mini Places dataset. The dataset consists of 120K images partitioned into three sets: training (100K), validation (10K), and testing (10K). Performance in the Mini Places Challenge is

Structure	Evaluation Method	Top-1 Error	Top-5 Error
Refnet	—	64.5%	35.1%
Shallow WW4-VGG	—	51.43%	22.3%
Deep WW4-VGG-A	—	51.16%	23.38%
Deep WW4-VGG-B	—	52.81%	23.66%
Shallow WW4-VGG	Multi-Crop	49.60%	18.99%
Deep WW4-VGG-A	Multi-Crop	47.16%	19.73%
Shallow and Deep-A nets	Combination and Multi-Crop	45.26%	17.78%
Shallow, Deep-A , Deep-B	Combination and Multi-Crop	45.06%	17.68%

Table 2. Results of model performances with various evaluation techniques. '–' denotes that no additional evaluation technique was used: classification is the result of the softmax layer.

Model	Refnet	WW4-VGG	Deep WW4-VGG
Parameters	2,687,873	3,728,640	8,448,256

Table 3. The number of parameters for each model.

evaluated with two metrics: top-1 and top-5 classification error. The accuracies listed below are achieved using the validation set as the test set.

5.1. Model Evaluation

Though we eventually yield our lowest error rate by combining different models and using data augmentation at test time, we begin by evaluating the accuracy of our three models independently when tested on only the original validation images scaled down to 100×100. All three models performed very similarly, each within 1% of one another. Shallow WW4-VGG had the best top-5 error at 22.3%, while Deep WW4-VGG-A had the best top-1 error at 51.43%. Table 3 shows a comparison of each net and its error rates. All models outperformed Refnet. In addition, our models performed similarly to VGG-16's performance on the full Places 2 dataset, which had a top-1 error of 52.4% [3].

5.2. Multiple Crop Evaluation

We now evaluate the performance of each net individually using the multi-crop test time technique described in 4.3. This technique significantly improved performance in all models. The top-5 error rates of Shallow WW4-VGG and Deep WW4-VGG-A models decreased by 3.31% and 3.65% respectively. The top-1 error rates decreased by 1.83% and 4% respectively.

5.3. Combining Nets

We gained a final boost in accuracy by using the outputs of multiple nets in conjunction to classify each image. A combination of WW4-VGG and Deep WW4-VGG-A, in addition to using multi-crop, classified scenes with a top-5 error of 17.78% and a top-1 error of 45.26%. Adding the Deep WW4-VGG-B net to make a three-net combination only led to a slight accuracy improvement of roughly 0.2% for both top-1 and top-5 error.

6. Future Work

It is possible that similar accuracy could be achieved with even fewer convolutional weight layers, an area which we intend to explore in future work.

In addition, We originally hoped to find a more intelligent way to combine the outputs of each net than simple averaging. One could potentially use the confusion matrix of each net to combine their outputs in a way that further compensated for each net's weaknesses. For example, our shallow net may be known to misclassify images of outdoor pools as lakes while our deep net does not. If our shallow net classifies a given test image as a lake and our deep net classifies it as a pool, we should weight the deeper net's prediction higher given what we know about our shallow net's misclassification tendencies. The most practical way to take advantage of this information would be to use an SVM to learn how the output of each net relates to the true classification of the image when run on the validation set. This poses a risk of overfitting on the validation set, but is a path we will explore in the future.

7. Conclusion

We successfully reduced the original VGG-16 structure to achieve a state of the art accuracy rate on the Mini Places Challenge dataset. We demonstrated that a deeper net is not necessarily a more accurate net by showing the similar error rates between our eight and ten layer models. This is most likely due to the risk of overfitting with additional deep layers, as our deep model had twice as many learning parameters as the shallow model.

We also showed the benefit of averaging the output of different structured nets to improve classification. Though our deep and shallow models had similar error rates, their error rates differ on a category-by-category basis, showing that they have distinct strengths and weaknesses. The top-

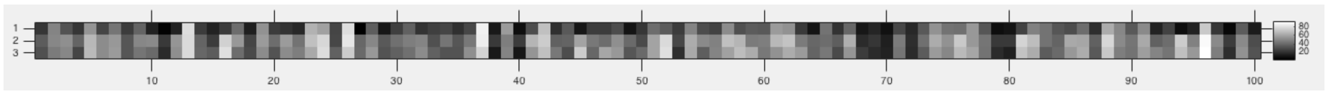

Figure 3. Each cell represents the rate at which one of our three models correctly classifies images in the category denoted by the column. Top down, the rows are WW4-VGG, Deep WW4-VGG-A, Deep WW4-VGG-B. A brighter cell represents a more frequently accurate classification of that scene category.

1 error rates for each model and each scene category are shown in Figure 3. This set of differing strengths and weaknesses is what makes our shallow and deep net combination so powerful, as it capitalizes on the strengths of two nets while allowing them to compensate for one another's weaknesses. Figure 3 also demonstrates why the three-net combination did not gain an advantage in performance: Deep WW4-VGG-A and Deep-WW4-VGG-B have near identical error rates for each category. Lastly, our nets in comparison to Refnet achieve a much higher accuracy without a dramatic increase in learning parameters, further demonstrating the advantage of a consistently small kernel size.

References

[1] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich. "Going Deeper with Convolutions." *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[2] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. "Object Detectors Emerge in Deep Scene CNNs." *International Conference on Learning Representations (ICLR) oral*, 2015.

[3] B. Zhou, A. Khosla, A. Lapedriza, A. Torralba and A. Oliva. "Places2: A Large-Scale Database for Scene Understanding," *Arxiv*, 2015.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton. "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[5] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," in *International Conference on Learning Representations (ICLR)*, 2015.

[6] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. "Object Detectors Emerge in Deep Scene CNNs," in *International Conference on Learning Representations (ICLR) oral*, 2015.

[7] Y. Jia, "Caffe: An open source convolutional architecture for fast feature embedding." http://caffe.berkeleyvision.org/, 2015.