KamiNet—A Convolutional Neural Network for Tiny ImageNet Challenge

Shaoming Feng
Stanford University
superfsm@stanford.edu

Liang Shi
Stanford University
liangs@stanford.edu

Abstract

In this paper, we address the Tiny ImageNet classification challenge from the perspective of data selection and training. We evaluate the effectiveness of the training data and propose a method to filter the invalid and un-helpful samples. Different CNN architectures are constructed to verify the validity of proposed algorithms. Experiment result shows our data filtering scheme combined with an appropriate CNN structure can boost the classification result by at least 3.6% percent in current dataset. The general idea of our algorithm can be extended to other dataset with similar characteristics and help improve classification result.

1. Introduction

Tiny ImageNet Challenge[2] is a subset of ImageNet Challenge[3]. It contains a total of 100,000 images in 200 categories with object location information. Challengers are asked to predict the class label of each image with no necessity to localize the object. The challenge is evaluated by the classification accuracy on the test set, of which the ground truth labels are not unveiled to challengers.

Conventional approaches to image classification problem consist of two explicit stages — feature extraction and classifier training. Local dense features like SIFT(Lowe, 2004)[14], HoG(Dalal et al., 2005)[8] and LBP(Ahonen et al., 2006)[6] are usually extracted first. Global image representation is then applied to pool the local features, from naive concatenating and histogram to more complicated scheme like spatial pyramid matching[10], Fisher vector representation[11] and non-linear coding representations[15]. After that, a classifier is trained using the encoded representation, usually a linear or kernelled SVM, to give prediction on the test set. Sometimes, more than one classifiers are trained with different feature combination, and their weighted average result will be considered, which in most cases outperforms a single classifier. This pipeline can work quite well when the dataset is relatively small — on the order of tens of thousands of images. But for dataset like ImageNet, which consists of over 15 million labeled images in over 22000 categories, both feature discriminability and model capacity become unaffordable at that scale.

Recently, the deep convolution neural network (CNN), with its strong ability to capture the nature of image and variable learning capacity, offers a new perspective for solving large-scale image classification and recognition problem. CNN integrates the feature extraction and classifier training in one coherent structure and provides an end-to-end solution from raw pixel data to class scores. CNN architecture makes the explicit assumption that the input are images and revises the original neural network to be more efficient. Specifically, the neurons are arranged in 3 dimensions and only connected to a small region of its previous layer. To further reduce the number of neurons, it uses a parameter sharing scheme which applies same weight vector for all the neurons in a single depth slice.

In this report, we emphasize on the data preprocessing problem in CNN and propose a method to evaluate the quality of labeled training data. We develop a data selection scheme which can filter out invalid or un-helpful samples by examining the shape and size of the bounded object. We validate our method through controlled experiments and show that it can indeed improve the classification results.

We develop our CNN with the deep learning framework — Caffe[1]. We also obtained the help from NVIDIA[4] to access their cluster with Tesla K40 and K80s for training the model. The prediction is done on i7-4770K CPU. The size and computing speed of our network is limited by the Caffe framework, which does not support multi-GPU training currently. It is also limited by the restriction of python framework on server. The result can be improved by applying same techniques with state-of-art method.

2. Background

ImageNet challenge has been hold for five years. At the year of 2012, the emergence of CNN indicates a turning point for the large scale image classification problem. Here we briefly introduce the important work being done every year based on a summarization paper[12]. Many works in-

troduced here also provide helpful insights when we design our CNN structure.

The ImageNet challenge starts from 2010. Most models in that years use a combination of traditional local image features such as SIFT, HOG, etc and trained by a linear SVM. The winner comes from NEC team who used SIFT and LBP features wiht two non-linear coding representation and a stochastic SVM.

At 2011, the winner team XRCE, applying high dimensional image signatures with compression using product quantization and one-vs-all SVMs.

2012 was the most important year. The winner, SuperVision, trained a CNN with 60 million parameters. Dropout layers were added to the network. This determined the basic structure of the state-of-art CNN. Later, Visual Geometry Group (VGG) published a paper that analyzed all the tricks used in this network, which provided a very detailed explanation of the these techniques.[7]

A great step is made in 2014 by GoogLeNet[13], who achieved an astonishing 6.7% error rate for the classification task. They introduce a multi-scale idea with intuitions gained from the Hebbian principle. Addition dimension reduction layers allowed them to increase both the depth and width of the network with incurring significant computational overhead.

For our project, we focused on learning the techniques mainly from VGG and GoogleLeNet, as well as others, to develop our own network that fits the scale of our data and hardware.

3. The Dataset

Tiny ImageNet is a subset of the ImageNet dataset. It contains 100,000 images in JPEG format, with a dimension of 64×64. There are a total of 200 classes. Each class has 500 training images, 50 validation images, and 50 test images. Labels and bounding boxes are provided for the training and validation sets but not for the test set. All the information is stored in a text file along with the dataset. The challenger obtains their prediction accuracy by uploading the result to an evaluation server.

Before any data augmentation and model training, all the images is substracted by the mean pixel intensity of each channel. It ensures the model to be trained on centered raw RGB images. [9] Training data is converted to Lightning Database[5] (LMDB) format required by Caffe framework. Each image is paired with an index number, which is uniquely mapped to the string provided in the Tiny ImageNet dataset.

3.1. Invalid and Un-helpful Data

Without directly using the provided training data, we go through the dataset first and find there are some samples should not be included. Particularly, We found two categories of data needs to be removed: invalid data and unhelpful data.

Figure 1. Example of invalid data(a volleyball should be in the red box)

Figure 2. Example of un-helpful data. The basketball in red box can hardly be seen.

Figure 1 shows an example of invalid data. It is labeled with "volleyball", where the location of the object is (33,0)-(39,0) — the top-left corner and bottom-right corner of the bounding box. Geometrically, it is a 8×1 slit at the very top of the image. We examine the entire training set and notice the "1 pixel width" or "1 pixed height" bounding boxs appear frequently and can show up at anywhere in the image, whether the border or center. In most cases, the object inside cannot be distinguished at all. We believe it is caused by using an improper shrinking program to gener-

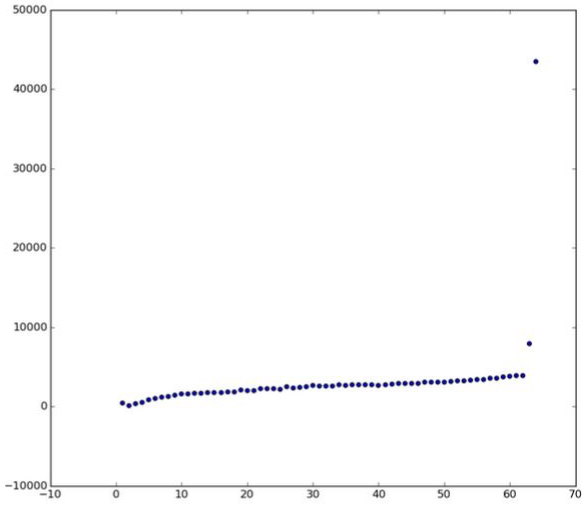

Figure 3. Bounding box edge size vs Number of images

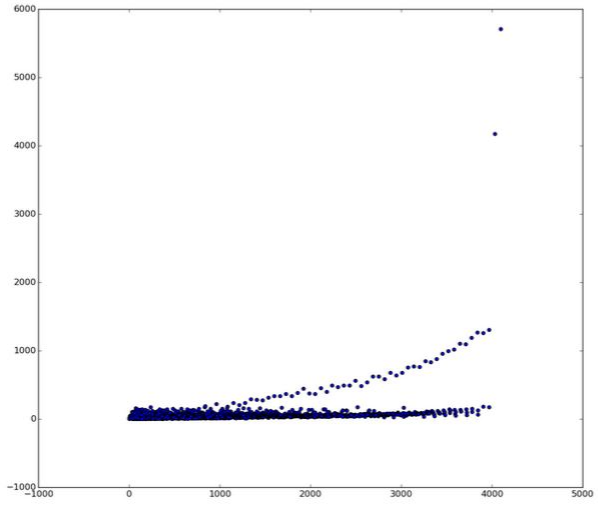

Figure 4. Bounding box area size vs Number of images

ate Tiny ImageNet dataset from the original high resolution ImageNet dataset. We denote this kind of images invalid because the object inside the bounding box does not match its label.

Figure 2 shows an example of un-helpful data. It is labeled with "basketball", and the bounding box is a 4×5 rectangular located at (29,23)-(32,27). Unlike the invalid image, the basketball is identifiable but requires extremely careful observation, for it is quite small and mixed into the noisy background caused by downsampling. For this image, "human", "sports" or "building" are more likely to be predicted if such labels exist as they occupy a much larger region. As a result, it is reasonable to make the assumption if bounding box of the object is smaller than certain threshold, it will not be helpful for generalizing the model of its corresponding class.

3.2. Bounding Box Distribution

Notice the invalid and un-helpful data has a very unbalanced length-to-width ratio or an extremely small area. In order to find a decision boundary for affirming the validity of sample, we first study the statistical distribution of bounding box on these two features.

In particular, we account three properties of the data. First is the distribution of a single side length of the bounding box(e.g., for a bounding box of 30×50, the number 30 and 50 will be counted separately). Second is the is the area size of the bounding box. Third is the length-to-width ratio. These three properties can exhaustively reflect the two features of the invalid and un-helpful data. If a discontinued joint exists at a proper location, we can use it as a threshold to perform filtering.

Figure 3,4 and 5 show these three distributions of the

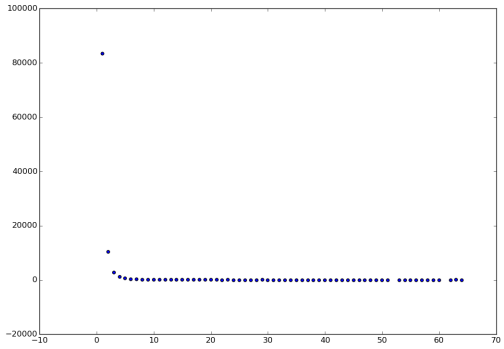

Figure 5. length-to-width ratio vs Number of images

training data. In Figure 3, bounding boxs with edge of 63 and 64 pixels take up about 1/4 of the entire set. (The total number of count is 200,000 based on the definition.). In Figure 4, 64×64 and 64×63 boxes together take up about 10% of the data. In the Figure 5, we can see that the images with a length-to-width ratio of 1:1 to 1:3 takes up about 90% of all the data.

Based on these three distribution graph, it is obvious that we can use the length-to-width ratio as the main distinguisher and use the bounding box size as the a subsidery constrain.

3.3. Data with Cropping

Sampling random crops is a classic way of addressing overfitting. When applying it to images with ground-truth object bounding box, one case we need to pay attention is the object being cut appart. Figure 6 shows an extreme case in which the cropped image preserve the minimum area of

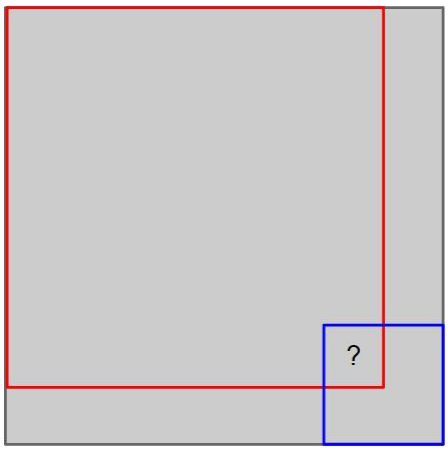

Figure 6. Red box is random crop. Blue box is the object location. How big should "?" area be?

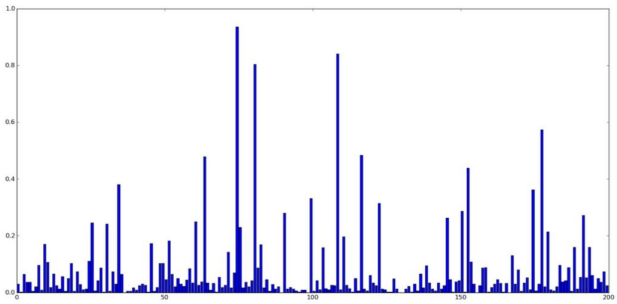

Figure 7. Percentage of data removed for each of the 200 class.

the object. As shown, the crop is taken on top left corner whereas the object is on bottom right, only the overlapped region can be learned by the CNN. In other cases, the shape of the bounding box can be cropped to a very unbalanced length-to-width ratio which also leads to an improper sample. Thus, all the filtering is processed after the data being cropped.

3.4. Threshold and result

So finally, based on crop size of 56×56 (we will explain why), we set the object size threshold to be 256, which is equal to 16×16 if the bounding box is a square. We also constrain the length-to-width ratio between 1:1 to 1:3.

To get an idea on how much data is removed for each class, we plotted Figure 7. It shows that for 3 out of the 200 classes, we removed nearly 80% to 95 % percent of the data. For 20 out of the 200 classes, more than 20% images from each classes are removed. We considered that this is an acceptable amount, taking into account of the accuracy we can reach for now (about 41%).

Since training a new model at this scale takes quite a long time, we are not able to fine tune all the filtering parameters. Based on current result, we believe the accuracy can be further improved with customized parameters.

Structure 1	Structure 2	Structure 3	KamiNet
Data	Data	Data	Data
			Filtering
		Crop + Flip	Crop + Flip
Conv3-32	Conv3-256	Conv3-256	Conv3-256
Conv3-32	Conv3-256	Conv3-256	Conv3-256
Conv3-32	Conv3-256	Conv3-256	Conv3-256
maxpool	maxpool	maxpool	maxpool
Conv3-64	Conv3-512	Conv3-512	Conv3-512
Conv3-64	Conv3-512	Conv3-512	Conv3-512
Conv3-64	Conv3-512	Conv3-512	Conv3-512
maxpool	maxpool	maxpool	maxpool
Conv3-128	Conv3-512	Conv3-512	Conv3-512
Conv3-128	Conv3-512	Conv3-512	Conv3-512
Conv3-128	Conv3-512	Conv3-512	Conv3-512
maxpool	maxpool	maxpool	maxpool
FC-1024	FC-4096	FC-4096	FC-4096
FC-1024	FC-4096	FC-4096	FC-4096
FC-200	FC-200	FC-200	FC-200
Softmax	Softmax	Softmax	Softmax

Table 1. KamiNet Structure and comparison with earlier version of KamiNet (ReLU after each Conv layer)

4. The Architecture

In this section, we introduce all the CNNs being trained during the development of KamiNet. To fully evaluate the effectiveness of our data filtering scheme, we set up a controlled experiment with four consecutively improved CNN structures. The structure 2 increases the depth of entire net and each Conv layers. The structure 3 augments the data with Crop+Flip scheme. Finally, KamiNet performs data filtering before data augmentation. All the structures are well designed after in depth study of related papers and consideration of our computational capacity.

4.1. Scale and Non-linearity of Model

Initially, we want to build our network as large as possible within the limit of graphical memory. However, one big defect of training our CNN versus the state-of-art ones is we can not utilize parallel computing through multiple GPUs, for the limitation of the server. As a result, we construct our model and training batch in a moderate size to keep a good balance between time cost and accuracy.

For the Conv layers in all the structures, we use 3×3 receptive field and bundle every three layers as a Conv stack. A stack of three 3×3 Conv layers has a 7×7 effective receptive field. However, comparing with a single 7×7 layer, it incorporates three non-linear rectification layers instead of

one, which makes the decision function more discrimitive. On the other hand, it also decreases the number of parameters in the model and improves the training efficency.

Starting from structure 2, we increase the depth of the volume of neuron to at least 256. Comparing to structure 1, it dramatically increases the features extracted at each layer and improve the capacity of mdoel. Results show that classification accuracy takes a big leap after this modification.

4.2. Crop and Flip

VGG paper "Return of the Devil in the detail"[7] suggests, random cropping and flipping will increase the validation accuracy greatly. Many papers have investigated the best cropping ratio and showed that a crop size as 7/8 of the original image gives most stable and relatively well result . So in our architecture, a size of 56×56 is used. We perform all the crops and filps randomly, with every possibility being choosed equally.

4.3. Training

For weight initialization, we set all the weight based on Gaussian distribution. $\mathcal{N}(\mu = 0, \sigma = 0.1)$ The bias term was initialized to 0 instead.

For hyperparameter, we tried VGG's value as well as a scale of 10 up and down. Then we figured that VGG's value is still the best one even we have different CNN structure. So here are the decisions we made. Each layer uses the same parameter setting. Mini-batch gradient descent, which is default one in caffe framework, is being applied. Batch size is set to 200, otherwise we will hit out-of-memory error. Momentus is set to 0.9 and weight decau is set to 5E-4, as a copy of VGG's net. Dropout regularization is set to 0.5, though we want to try less in the future since we believe we overfit our data slightly.

We scaled our learning rate at 18000th iteration and at 35000th iteration again by 0.1. This is where we observed that our validate accuracy doesn't increase anymore. We stopped the learning after 44 epochs.

5. Experiments

In this section, we present the image classification results achieved by the four structures. The results will be evaluated progressively by comparing the accuracy change between each structures. Since making prediction for the entire test set will take up to ten hours, we only predict the test set using KamiNet and obtain the test accuracy from the evaluation server. An overall results is showed in Table 3.

5.1. Structure 1 to Structure 2

Notice, both training and validation accuracy get hugh improve when the depth of the volume of neurons increases. This phenomenon shows the learning capacity of structure 1 is obviously unaffordable to the problem scale of Tiny ImageNet. The validation accuracy stops increasing at 5.6% and the training accuracy is 48.1% at that time. As a comparison, the structure 2 scales well to the problem and its training accuracy reaches to 98.1% at the end. The validation accuracy also reaches to 40.9% which is almost eight times to the structure 1.

	S1	S2	S3	KamiNet
Train Acc	48.1%	98.1%	96.7%	97.5%
Val Acc	5.6%	40.9%	43.5%	47.1%
Test Acc				49.5%

Table 2. Classification Result

Considering the size and variaty of images in Tiny ImageNet, the increased volume depth can capture more information like various oriented edges, the blob of colors and distinguishable patterns. All these features offer a more distinctive description of the region neurons connected to and consequently increase the capacity of the entire model. It is obvious that if we keep increasing the layers in each Conv stack, the validation accuracy can be further imrpoved.

5.2. Structure 2 to Structure 3

There is a noticeable increase after performing the data augmentation. The validation accuracy increases from 40.9% to $43,5\%$. This improvement is achieved without modify the propagation layers. Notice, the training accuracy of both structures has reached nearly 100% percent. In most cases, it is an indication of overfitting. Flipping and cropping help increase the number of training samples artificially and without actually increase the training data in the input. Rather, we just crop and flip the images at every batch iteration. As shown in the table, the augmented data slightly mitigates the possible overfitting. There are many other ways to perform data augment, for example, modifying the image contrast, making random tint. We believe the validation accuracy can be further improved if we also apply these data augmentation techniques.

5.3. Structure 3 to KamiNet

The KamiNet takes the validation accuracy to a higher level with an increasing from 43.5% to 47.1. The improvement is even greater than the extent achieved from structure 2 to 3. The results verify our assumption that removing the invalid and un-helpful data will improve the accuracy of model. Notice, the training accuracy of KamiNet is slightly increased than S3, it is probability caused by removing about 1/10 data in the original training set. Since most of the removed data is ill-labelled, not only the data scale but also the intraclass variation decreases. Thus, it is reasonable to see the increase of training accuracy. The next section will give a more detailed analysis based on accuracy

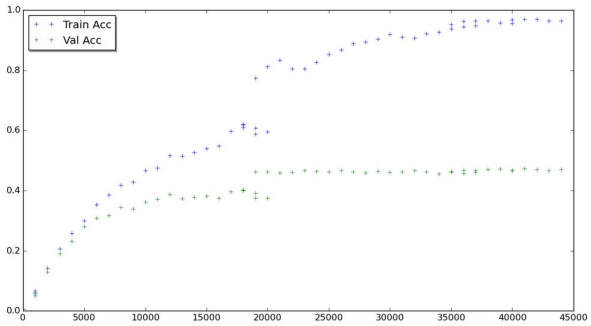

Figure 8. Training and Accuracy of KamiNet

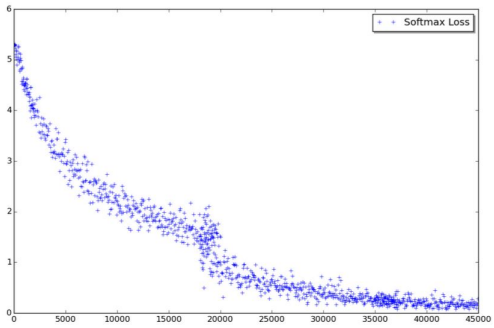

Figure 9. Lost vs Iteration of KamiNet

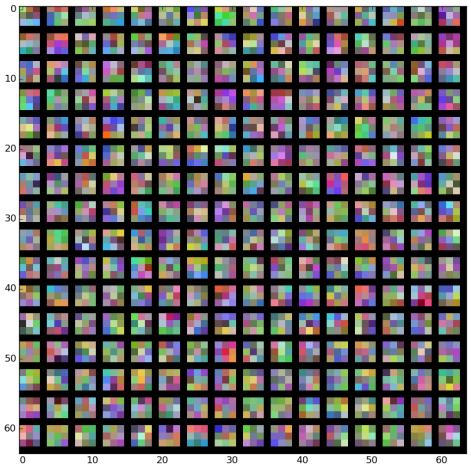

Figure 10. Weights of KamiNet's first layer

and lost curve.

5.4. Accuracy and Lost

Figure 8 shows the plot for training and validation accuracy, of KamiNet. And Figure 9 plots the softmax loss

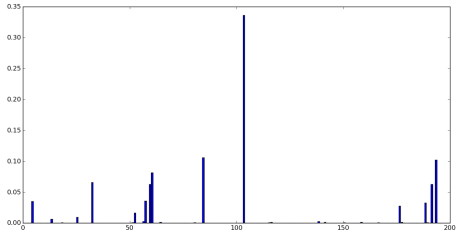

Figure 11. Classification result of a single test image

based on iterations.

As we can see obviously from the plot, we scaled our learning rate by 0.1 at iteration 18000 and 35000. The reason we scale the learning rate there was because the validation accuracy doesn't increase anymore at that point. So further training will overfit the data.

From the loss plot, we can see that it has a good shape. It hasn't converged yet based on loss graph, even though the validation accuracy doesn't increase anymore. From the accuracy plot, validation accuracy only increases a little bit after scaling learning rate down. Since the training accuracy was increasing, we'd say we overfit it a little bit. But this is not too bad because it is not decreasing obviously.

We considered that the difference between our model and state-of-art model is that our module is not as big as VGG's or GoogLeNet. Also, another reason for the difference is we are missing techniques such as channel switching and other augmentation method.

5.5. Weights and Prediction

Figure 10 shows a weight of depth 256, for a single reception field of 3 x 3. We can see they don't look the same, which means we broke the symmetricity successfully. We cannot find any meaningful patterns because the reception field is very small.

Figure 11 shows the predicted probability for the first picture. The class with highest probability is "candle, taper, wax light". As it is an image from test set, we don't have the exact label for it. It looks like a bottle. We think this is reasonable because they all have a long and smooth shape.

6. Conclusion

The final KamiNet structure can achieve an error rate of 50.5

The data filtering method we used, which validates the bounding box shape and length-to-width ratio can help improve the validation accuracy by 3.6%. In the case of Tiny ImageNet, we considered this to be a good result.

All the techniques we learned from different papers provided great help in terms of designing the network and tuning hyperparameters. We believed that by applying more

state-of-art techniques as well as accessing the hardware with fewer limit, our accuracy would be much better.

The future works of our project includes tuning various data filtering threshold, trying different strategy, applying latest techniques, having larger model capacity, and utilizing the multi-GPU hardware better. Our next step is to try the origional ImageNet challenge or try add location detection functionality in our mode.

References

[1] http://caffe.berkeleyvision.org/.

[2] https://tinyimagenet.herokuapp.com/.

[3] http://www.image-net.org/.

[4] http://www.nvidia.com.

[5] symas.com/mdb/.

[6] T. Ahonen, A. Hadid, and M. Pietikainen. Face description with local binary patterns: Application to face recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(12):2037–2041, 2006.

[7] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. *arXiv preprint arXiv:1405.3531*, 2014.

[8] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. 1:886–893, 2005.

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks, 2012.

[10] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 2169–2178. IEEE, 2006.

[11] F. Perronnin and C. Dance. Fisher kernels on visual vocabularies for image categorization. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[12] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *arXiv preprint arXiv:1409.0575*, 2014.

[13] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *arXiv preprint arXiv:1409.4842*, 2014.

[14] K. E. Van De Sande, T. Gevers, and C. G. Snoek. Evaluating color descriptors for object and scene recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(9):1582–1596, 2010.

[15] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong. Locality-constrained linear coding for image classification. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 3360–3367. IEEE, 2010.