Transfer Learning for Deep Convolutional Neural Network on Food Image Dataset

Abstract

Deep Convolutional Neural Network (CNN) has recently achieved great successes in object recognition competitions, many of which have been accomplished using AlexNet-like architectures. Meanwhile, the explorations of the new born GoogLeNet remain insufficient and little work has been done to compare two different deep C-NN architectures. Moreover, transfer learning with deep CNN has shown impressive results on benchmark datasets and can be an efficient way for specific recognition tasks. In order to explore these two problems, we first apply transfer learning for food images recognition task. The results show that incorporating ImageNet dataset in pre-training can lead to improved performance over using only target labeled dataset. We also conduct empirical studies comparing the performance of AlexNet and GoogLeNet on this task. We reveal that the small receptive field used in GoogLeNet can improve both training and computational efficiency for transfer learning. It is also shown in our experiments that deep CNN has strong generalization ability in transfer learning across two food datasets with little overlap categories even though the target dataset contains just a few labeled instances per category.

1. Introduction

Recognizing the objects in the world is the most fundamental function for human to understand the world the whole procedure of which only takes a few tens of milliseconds for human brain. Recently, Convolutional Neural Network (CNN) shows its potential to replace the human engineered features, such as SIFT (Lowe, 1999), SURF (Bay et al., 2006) and HOG (Dalal & Triggs, 2005) etc, in real object recognition tasks. The success of CNNs on large scale image set started from Krizhevsky et al and their 8 layer model AlexNet in 2012 ImageNet Large-Scale Visual Recognition Challenge (ILSVRC2012), reaching a top-5 accuracy at 83% (Krizhevsky et al., 2012). Since then, many attempts have been made to improve the architecture of Krizhevsky. By reducing the size of the receptive field and stride, Zeiler and Fergus improve AlexNet by 1.7% on top 5 accuracy (Zeiler & Fergus, 2014). With the help of high performances computing systems, such as GPU and large scale distributed clusters, it is possible for researchers to explore larger and more complex architecture. By both adding extra convolutional layers between two pooling layers and reducing the receptive window size, Simonyan and Zisserman built a 19 layer very deep CNN and achieved 92.5% top-5 accuracy (Simonyan & Zisserman, 2014).

After the AlexNet-like deep CNNs conquered ILSVR-C2012 and ILSVRC2013, Szegedy et al built a 22 layers deep network, called GoogLeNet and won the 1st prize on ILSVRC2014 for 93.33% top-5 accuracy, almost as good as human annotation(Szegedy et al., 2014). Different from AlexNet-like architecture, GoogLeNet shows another trend of designing deep CNN with many 1×1 receptive field. However, its unique architecture has not been widely studied and few result can be found for applying GoogLeNet on other benchmark datasets.

Unlike the local features such as SIFT or SURF, which present an intuitive interpretation of spatial property, we still don't have enough knowledge to fully understand the visual features of CNN learned in each layer. However, the truth that deep CNN outperforms other shallow models by a large margin in some real image recognition tasks encourages researchers to build deep architecture with powerful high performance hardware and larger datasets. With the help of high performance GPU clusters and data argumentation, people are more enthusiastic to explore bigger networks on complex recognition problems without much interpretation which makes other researchers difficult to understand the advantages of these architectures and explore their own ones.

Since these CNN models are trained on a very large image dataset and can learn hierarchical features, they have strong generalization ability and can be applied in many other scenarios. Applying the pre-trained model from ImageNet dataset on other object recognition benchmark datasets shows some impressive results. Zeiler et al. ap-

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

plied their pre-trained model on Caltech-256 with just 15 instances per class to fine-tune the model and improved the previous state-of-the-art in which about 60 instances are used for training, by almost 10% (Zeiler & Fergus, 2014). Chatfield et al used their pre-trained model on VOC2007 dataset and outperformed the previous state-of-the-art by 0.9% (Chatfield et al., 2014). Few results have been shown related to the transfer learning performance of deep CNN on a more challenging and specific real world recognition problem.

In this paper, we apply two kinds of deep CNN model, AlexNet and GoogLeNet, on a specific real recognition problem, food recognition, and discuss some tricks in fine-tuning the existing CNN architectures on this problem. To our best knowledge, little work has done to discuss the architecture of GoogLeNet while the architecture of AlexNet has been widely studied and improved. Also, no work has been found to compare two deep CNNs with different architectures. By comparing some statistics of the weights and neuron responses of these two architectures, we find that the 1×1 small receptive field used in GoogLeNet can improve both computational and training efficiency which leads to its success. Also, we conduct several experiments to stimulate a real world scenario when the training labeled data is rare. The results reveal that deep CNN can work well while transferring knowledge from general recognition task to specific one in this scenario. We achieve 95% accuracy trained on full dataset while just utilizing half of the dataset.

The rest of this paper is organized as follow: in Section 2, the two food image datasets and two deep CNN architectures are introduced. In Section 3, some experimental results are shown and we also compare the performance between the deep CNNs as well as some traditional methods on these two datasets. And some discussion of the Inception's architecture and statistics are shown in Section 3. We also show some fine-tuning results when the training examples are rare for each class.

2. Experimental Setup

In this section, we introduce some details about the food datasets and the two architectures used for our experiments.

2.1. Models

In this paper, AlexNet and GoogLeNet are their Caffe (Jia et al., 2014) implementation and all the results for a specific CNN architecture are obtained from single model.

AlexNet contains 5 layers followed by the auxiliary classifier which contains 2 fully connected layers (FC) and 1 softmax layer. Each of the first two layers can be subdivided into 3 components: convolutional layer with rectified

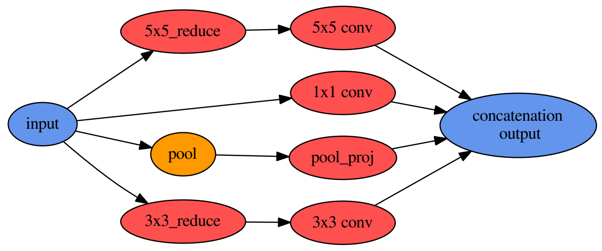

Figure 1. Inception Cell. $n \times n$ stands for size n receptive field, $n \times n\_reduce$ stands for the 1×1 convolutional layer before the $n \times n$ convolution layer and $pool\_proj$ is another 1×1 convolutional layer after the MAX pooling layer. The output layer concatenate all its input layers.

linear units (ReLUs), local response normalization layer (LRN) and max pooling layer. Layer 3 and layer 4 contain just convolutional layer with ReLUs while layer 5 is similar to the first two layers except for the LRN. For each of the fully connected layer, 1 ReLUs and 1 dropout (Srivastava et al., 2014) layer are followed.

GoogLeNet shows another trend of deep CNN architecture with lots of small receptive fields. Figure 1 shows the architecture a inception cell. Inspired by (Lin et al., 2013), lots of 1×1 convolutional layers are used for computational efficiency. Another interesting feature of GoogLeNet is that there are two extra auxiliary classifiers in intermediate layers. During the training procedure, the loss of these two classifiers are counted into the total loss with a discount weight 0.3, in addition with the loss of the classifier on top. More architecture details can be found from (Szegedy et al., 2014).

2.2. Food Datasets

Besides ImageNet dataset, there are many popular benchmark datasets for image classification tasks such as Caltech dataset and CIFAR dataset, which contain hundreds of classes. However, in this paper, we try to focus on a more specific area, food classification. Compared to other classification tasks, there are some properties of the food (dishes) which make the tasks become a real challenge: i)food doesn't have any distinctive spatial layout: for other tasks like scene recognition, we can always find some discriminative features such as buildings or trees, etc. ii) food class is a small sub-category among all the categories in daily life, so the inter-class variation is relatively small; on the other hand, the contour of a food varies depending on many aspects such as the point of the view or even its components. So these properties make food classification catastrophic for some recognition algorithms. Therefore, the training these two models on food classification task can reveal some important aspects of themselves and help us better understand their architectures. In this paper, we used two image datasets Food-256 (Kawano & Yanai, 2014)[1] and Food-101 (Bossard et al., 2014)[2]. It is worthy to mention that PFID dataset is also a big public image database for classification, but their images are collected in a laboratory condition which is considerably not applicable for real recognition task.

Food-256 Dataset. This is a relatively small dataset containing 256 kinds of foods and 31644 images from various countries such as French, Italian, US, Chinese, Thai, Vietnamese, Japanese and Indonesia. The distribution among classes is not even and the biggest class (vegetable tempura) contains 731 images while the smallest one contains just 100 images. For this "small" dataset, we randomly split the data into training and testing set, using around 80% (25361 images) and 20%(6303 images) of the original data respectively and keep the class distribution in these two sets uniform. The collector of this dataset also provides boundary box for each image to separate different foods and our dataset is cropped according to these boundary boxes.

Food-101 Dataset. This dataset contains 101-class real-world food (dish) images which were taken and labeled manually. The total number of images is 101,000 and there are exactly 1000 images for each of the class. Also, each class has been divided into training and testing set containing 750 images and 250 images respectively by its collector. The testing set is well cleaned manually while the training set is not well cleaned on purpose. This noisy training set is more similar to our real recognition situation and it is also a good way to see the effect of the noise on these two architectures.

Data argumentation is an efficient way to enrich the data. There are also some techniques that can applied to enlarge the dataset such as subsampling and mirroring. The original images are firstly resized to 256 $\times$ 256 pixels. We crop the 4 corners and center for each image according to the input size of each model and flap the 5 cropped images to obtain 10 crops. For the testing set, the prediction of an image is the average prediction of the 10 crops.

3. Experimental Discuss

Training a CNN with millions of parameters on a small dataset could always lead to horrible overfitting. But the idea of supervised pre-training on some huge image datasets could preventing this problem in certain degree. Compared to other initialized strategies according to certain distributions, the pre-trained model is initialized according to the distribution of the specific task. Indeed, this initialization has certain bias as there is no single dataset including all the invariance for natural images (Agrawal et al., 2014), but this bias could be reduced as the pre-trained image dataset increases and the fine-tuning can be benefit from this initialization.

[1]Dataset can be found http://foodcam.mobi/dataset.html

[2]Dataset can be found http://www.vision.ee.ethz.ch/datasets_extra/food-101

3.1. Pre-training and Fine-tuning

We conduct several experiments on both architectures and use different training initialization strategies for both Food-256 and Food-101 datasets. The scratch models are initialized with Gaussian distribution for AlexNet and Xavier algorithm (Glorot & Bengio, 2010) for GoogLeNet, which automatically determines the scale of initialization based on the number of input and output neurons. These two initializations are used for training the original models for the ImageNet task. The pre-trained and fine-tune models are initialized with the weights trained from ImageNet. For the pre-trained models, we just re-train the softmax layers while all the layers fine-tuned for the fine-tune models.

Table 1. Top-5 Accuracy in percent on fine-tuned, pre-trained and scratch model for two architectures

	AlexNet		GoogLeNet	
	Food-101	Food-256	Food-101	Food-256
Fine-tune	**88.12**	**85.59**	**93.51**	**90.66**
Pre-trained	76.49	79.26	82.84	83.77
Scratch	78.18	75.35	90.45	81.20

From Table 1 we can see that fine-tuning from pre-trained model can improve the performance of the CNN for our task. Compared to other traditional computer vision methods, GoogLeNet and AlexNet improve at least 27.35% and 15.64% respectively (see Table 2). Considering about the noisy images in Food-101 dataset, this 78.11% accuracy using fine-tuned GoogLeNet is good enough for a real recognition task. Also it is the state-of-the-art performance on this dataset. Because no one has shown any classification result on Food-256 dataset, our result could be a competitive baseline for this dataset.

In Figure 2 we visualize the responses of the pre-trained GoogLeNet model and fined-tuned GoogLeNet model with the same input image for some layers. We can see that the responses of the lower layer are similar as the lower level features are similar for most recognition tasks. Then we can see that the responses of the high-level neurons are different which leads to totally different recognition results. Since only the last layer (auxiliary classifier) of the pre-trained model is optimized, we can infer that the higher level features are more important which is consistent with

Table 2. Accuracy compared to other methods on Food-101 dataset

	RFDC(Bossard et al., 2014)	MLDS(≈(Singh et al., 2012))	GoogLeNet	AlexNet
Top1 accuracy	50.76%	42.63%	**78.11%**	66.40%

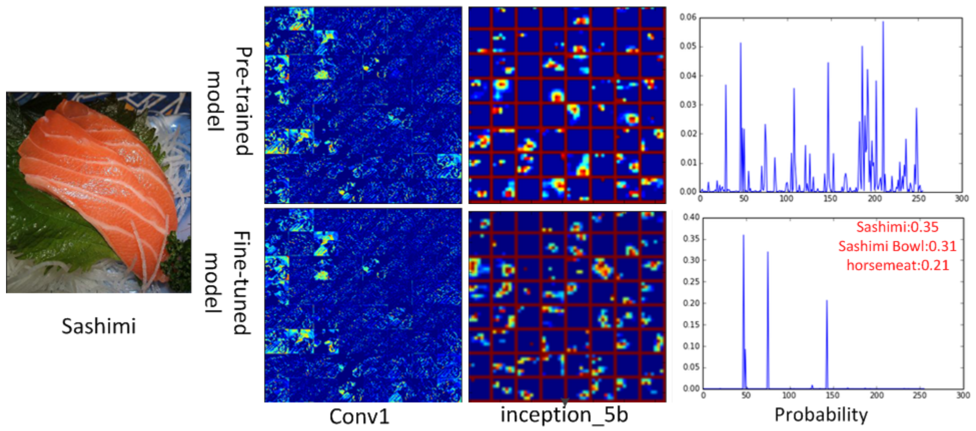

Figure 2. Visualization of some responses of different GoogLeNet models in different layers for the same input image. 64 neuron responses of each layer are shown. Conv1 is the first convolutional layer and Inception_5b is the last convolutional layer.

our intuition. Also from Table 1, it is interesting to see that for the Food-101 task, the accuracy of the scratch models outperforms the pre-trained models. Since Food-101 is a relatively large dataset with 750 images per class while Food-256 dataset is an imbalanced small one, this indicates that it is difficult to obtain a good deep CNN model while the data is insufficient.

From Table 1 we can see that GoogLeNet always performances better than AlexNet on both datasets. This implies that the higher level features of GoogLeNet are more distinctive compared to AlexNet and this is due to the special architecture of its basic unit, Inception. Table 3 and 4 show the weights' cosine similarity of each layer between the fine-tuned models and their pre-trained models. From the results we can see that the weights in the low layer are more similar which implies that these two architectures can learn the hierarchical features. As the low level features are similar for most of the tasks, the difference of the objects is determined by high-level ones which are the combination of these low level features. Also from Table 4, we can observe that, the weights of the pre-trained and fine-tuned models are extremely similar in AlexNet . This can be caused by two reasons:

- Size of receptive filed. Since ReLUs are used in both architectures, vanishing gradients do not exist. Rectified activation function is mathematically given by:

$$h = \max(w^T x, 0) = \begin{cases} w^T x & w^T x > 0 \\ 0 & else \end{cases} \qquad (1)$$

The ReLU is inactivated when its input is below 0 and its partial derivative is 0 as well. Sparsity can boost the performance of the linear classifier on top, but on the other hand, sparse representations make the network more difficult to train as well as fine-tuning. The derivative of the filter is $\frac{\partial J}{\partial w} = \frac{\partial J}{\partial y}\frac{\partial y}{\partial w} = \frac{\partial J}{\partial y} * x$ where $\frac{\partial J}{\partial y}$ denotes the partial derivative of the activation function, $y = w^T x$ and x denotes the inputs of the layer. The sparseness of the input could lead to sparse filter derivative for back propagation. Therefore, the filters of the fine-tuned AlexNet is extremely similar. Compared to large receptive field used in AlexNet, the inception in GoogLeNet employs 2 additional $n \times n\_reduced$ convolutional layers before the 3×3 and 5×5 convolutional layers (see Figure 1). Even though the most critical purpose of these two 1×1 convolutional layer is for computational efficiency, these 2 convolutional layers tend to squeeze the sparse input and generate a dense outputs as the input of the following layer. This makes the filters in the following layer more easily to be trained for transfer learning and generate efficient sparse representations.

- The pooling strategy. In AlexNet, max pooling is applied to all the pooling layers between several convolution layers and in back propagation, the max pooling layer always passes the error to the place where it came from. Since it only came from one place of the receptive field, the back propagation error is sparse and keeps the most filters' weights unchanged. In GoogLeNet, even though, there is a max pooling layer within every inception, there are other 3 back propagation errors, from $5 \times 5\_reduce$ and $3 \times 3\_reduce$ that can affect the weights of the previous inception.

3.2. Training across the datasets

From the previous experiments we can see that pre-training on the ImageNet dataset can improve the performance of the deep convolutional neural network on our specific area. In this part, we will discuss the generalization ability within the food recognition problem. Zhou et al. trained AlexNet for Scene Recognition across two datasets with identical categories (Zhou et al., 2014). But for more complex situation, such as two similar datasets with a little overlapped categories, we are very interested in exploring whether deep CNN can still successfully handle. Therefore, we conduct the following experiment to stimulate a more complex real world problem: transferring the knowledge from the fine-tuned Food-101 model to a target set, Food-256 dataset and continue fine-tune the model on it. To make the experiment more practical and complex, we limit the number of samples per category from Food-256 for training, because in practise, if we want to build a our own model from Deep CNN, the resource is limited and it is exhausted to collect hundreds of labeled images for each category.

The Food-101 and Food-256 datasets share about 46 categories of food even though the images in the same category may vary across these two datasets. The types of food in Food-101 are mainly western style while most types of food in Food-256 are typical Asian foods. We compare the top-5 accuracy of different size of subset for Food-256 on different pre-trained model and the results are shown in Table 6. The ImageNet columns denote the pre-trained model trained only on ImageNet images and the Food101_ft columns denote the pre-trained model trained on ImageNet images and then fine-tuned on Food-101.

From the result of Table 6 we can see that, with this second round of transfer learning, both CNNs can achieve around 95% of the accuracy trained on full dataset while just using about half of them (50 per class, 12800 of 25361 images). This indicates that when there is not enough labeled data, with its strong generalization ability, deep CNN trained from general task can still achieve satisfying result and perform even better when an additional relevant dataset is involved. This encouraging result may attract more people to use deep CNN for their specific task and continue to explore the potential of the existing architecture as well as designing new ones.

4. Conclusion

In this paper, we compare two different deep convolutional neural network architectures and their transferring ability on food datasets. Both architectures have shown their potential on generalization ability and we provide the-state-of-the-art on Food-101 dataset and competitive baseline on Food-256 dataset using fine-tuned GoogLeNet. GoogLeNet shows its strong ability on transferring the knowledge between different tasks with the help of the specially designed unit, Inception. We find that not only does the intensively used 1×1 convolutional layer in Inception reduce the computational cost, but it also helps to train the filters in the following layer. The filters after the 1×1 convolutional layer can be trained more efficiently while the 1×1 convolutional layer provides squeezed input and this feature helps GoogLeNet learn more complex high-level features. Moreover, in a more practical situation such as transfer learning within a specific area with a few labeled instances for the target set, both of these two architectures show some encouraging result in transferring knowledge across the dataset, reaching around 95% of the accuracy trained on full dataset with just half data.

Acknowledgments

References

Agrawal, Pulkit, Girshick, Ross, and Malik, Jitendra. Analyzing the performance of multilayer neural networks for object recognition. In *Computer Vision–ECCV 2014*, pp. 329–344. Springer, 2014.

Bay, Herbert, Tuytelaars, Tinne, and Van Gool, Luc. Surf: Speeded up robust features. In *Computer vision–ECCV 2006*, pp. 404–417. Springer, 2006.

Bossard, Lukas, Guillaumin, Matthieu, and Van Gool, Luc. Food-101 – mining discriminative components with random forests. In *European Conference on Computer Vision*, 2014.

Chatfield, K., Simonyan, K., Vedaldi, A., and Zisserman, A. Return of the devil in the details: Delving deep into convolutional nets. In *British Machine Vision Conference*, 2014.

Dalal, Navneet and Triggs, Bill. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pp. 886–893. IEEE, 2005.

Glorot, Xavier and Bengio, Yoshua. Understanding the difficulty of training deep feedforward neural networks. In *International conference on artificial intelligence and statistics*, pp. 249–256, 2010.

Jia, Yangqing, Shelhamer, Evan, Donahue, Jeff, Karayev, Sergey, Long, Jonathan, Girshick, Ross, Guadarrama,

Table 3. Cosine similarity of the inceptions between fine-tuned models and scratch model for GoogLeNet

food256						
	1x1	3x3_reduce	3x3	5x5_reduce	5x5	pool_proj
inception_3a	0.72	0.72	0.64	0.67	0.73	0.69
inception_3b	0.59	0.64	0.53	0.70	0.60	0.56
inception_4a	0.46	0.53	0.54	0.50	0.67	0.38
inception_4b	0.55	0.58	0.63	0.52	0.69	0.41
inception_4c	0.63	0.64	0.63	0.57	0.68	0.52
inception_4d	0.60	0.62	0.60	0.58	0.68	0.50
inception_4e	0.60	0.61	0.67	0.61	0.68	0.50
inception_5a	0.51	0.53	0.58	0.48	0.60	0.39
inception_5b	0.40	0.44	0.50	0.41	0.59	0.40
food101						
	1x1	3x3_reduce	3x3	5x5_reduce	5x5	pool_proj
inception_3a	0.71	0.72	0.63	0.67	0.73	0.68
inception_3b	0.56	0.63	0.50	0.71	0.60	0.53
inception_4a	0.43	0.50	0.50	0.47	0.62	0.36
inception_4b	0.48	0.52	0.57	0.50	0.67	0.35
inception_4c	0.57	0.61	0.59	0.53	0.63	0.47
inception_4d	0.54	0.58	0.53	0.54	0.64	0.44
inception_4e	0.53	0.54	0.61	0.55	0.62	0.42
inception_5a	0.43	0.47	0.53	0.45	0.57	0.34
inception_5b	0.36	0.39	0.46	0.38	0.52	0.37

Table 4. Cosine similarity of the layers between fine-tuned models and scratch model for AlexNet

	conv1	conv2	conv3	conv4	conv5	fc6	fc7
food256	0.997	0.987	0.976	0.976	0.978	0.936	0.923
food101	0.996	0.984	0.963	0.960	0.963	0.925	0.933

Table 5. Sparsity of the output for each unit in GoogLeNet inception for training data from Food101 in percent

	1x1	3x3_reduce	3x3	5x5_reduce	5x5	pool_proj
inception_3a	69.3 ± 1.3	69.6 ± 1.1	80.0 ± 1.0	64.1 ± 2.2	75.8 ± 1.6	76.2 ± 5.4
inception_3b	92.8 ± 0.9	76.5 ± 0.9	94.7 ± 0.9	71.6 ± 2.3	94.4 ± 0.5	94.7 ± 1.6
inception_4a	90.9 ± 0.9	70.0 ± 1.2	93.8 ± 1.1	63.3 ± 4.0	91.9 ± 1.8	95.1 ± 2.0
inception_4b	71.9 ± 1.6	67.5 ± 1.2	75.4 ± 1.0	58.5 ± 2.6	78.9 ± 1.6	85.6 ± 3.6
inception_4c	75.1 ± 2.4	72.6 ± 1.3	81.0 ± 2.0	66.3 ± 6.1	79.7 ± 3.6	88.1 ± 3.3
inception_4d	87.3 ± 2.7	78.0 ± 2.2	88.0 ± 1.6	67.9 ± 3.1	88.9 ± 2.8	93.0 ± 2.2
inception_4e	91.8 ± 1.1	62.3 ± 2.2	91.0 ± 2.5	49.5 ± 3.7	94.0 ± 1.0	92.3 ± 1.5
inception_5a	78.7 ± 1.6	66.5 ± 1.7	82.3 ± 2.6	59.9 ± 3.2	86.4 ± 2.3	87.1 ± 2.6
inception_5b	88.2 ± 2.3	86.8 ± 1.6	83.3 ± 4.4	84.0 ± 3.1	81.4 ± 5.3	94.7 ± 1.5

Table 6. Top5 Accuracy for transferring from Food101 to subset of Food256

	AlexNet		GoogLeNet	
instances per class	ImageNet	Food101_ft	ImageNet	Food101_ft
20	68.80	**75.12**	74.54	**77.77**
30	73.15	**77.02**	79.21	**81.06**
40	76.04	**80.23**	81.76	**83.52**
50	78.90	**81.66**	84.22	**85.84**
all	85.59	**87.21**	**90.66**	**90.65**

Sergio, and Darrell, Trevor. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the ACM International Conference on Multimedia*, pp. 675–678. ACM, 2014.

Kawano, Y. and Yanai, K. Automatic expansion of a food image dataset leveraging existing categories with domain adaptation. In *Proc. of ECCV Workshop on Transferring and Adapting Source Knowledge in Computer Vision (TASK-CV)*, 2014.

Krizhevsky, Alex, Sutskever, Ilya, and Hinton, Geoffrey E. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012.

Lin, Min, Chen, Qiang, and Yan, Shuicheng. Network in network. *CoRR*, abs/1312.4400, 2013.

Lowe, David G. Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pp. 1150–1157. Ieee, 1999.

Simonyan, Karen and Zisserman, Andrew. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

Singh, Saurabh, Gupta, Abhinav, and Efros, Alexei A. Unsupervised discovery of mid-level discriminative patches. In *Computer Vision–ECCV 2012*, pp. 73–86. Springer, 2012.

Srivastava, Nitish, Hinton, Geoffrey, Krizhevsky, Alex, Sutskever, Ilya, and Salakhutdinov, Ruslan. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1): 1929–1958, 2014.

Szegedy, Christian, Liu, Wei, Jia, Yangqing, Sermanet, Pierre, Reed, Scott, Anguelov, Dragomir, Erhan, Dumitru, Vanhoucke, Vincent, and Rabinovich, Andrew. Going deeper with convolutions. *arXiv preprint arXiv:1409.4842*, 2014.

Zeiler, Matthew D and Fergus, Rob. Visualizing and understanding convolutional networks. In *Computer Vision–ECCV 2014*, pp. 818–833. Springer, 2014.

Zhou, Bolei, Lapedriza, Agata, Xiao, Jianxiong, Torralba, Antonio, and Oliva, Aude. Learning deep features for scene recognition using places database. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N.D., and Weinberger, K.Q. (eds.), *Advances in Neural Information Processing Systems 27*, pp. 487–495. Curran Associates, Inc., 2014.