See-Flower: Fine-Grained Flower Image Classification Using Deep Transfer Learning

Qi Qu
University of California,
Los Angeles
qqu0127@cs.ucla.edu

Abstract

In this paper, we propose an approach to small sample, fined-grained image classification task using deep neural networks. Taking the idea from transfer learning, we extensively experiment tuning pretrained deep convolutional neural network on Oxford 102 Flower dataset achieve state-of-art result with a top-1 accuracy 90.4%.

In order to validate the model and further interpret the classifier, we apply a few visualization techniques to show how the trained deep convolutional network "see flowers" (how the trained model perceive the image data and how the decision is made) in our experiments. Code is available on https://github.com/qqu0127/see-flowers.

1. Introduction

Convolutional neural network (CNN) has been seen deeper and larger in recent years and achieved great performance on image classification tasks [1, 2]. A large amount of training data is essential to feed the huge network for visual recognition task and it consume great much amount of computational resources, usually done with computer clusters or at least with certain hardware support [3]. Though the results of deep networks lead the records of many public datasets, training a deep network with insufficient data usually leads to less promising results compared to traditional models or using manually picked features. There are also some common arguments on the lack of transparency and interpretability of the deep convolutional neural network, as it's more like a black box compared to traditional models, and hard to explain.

Lots of research has been done in general object classification with a variety of classes, while fine-grained classification received less attention, where the insufficiency of training data makes it hard to fully train a deep neural network from scratch. Solving the overfitting problem in such tasks is challenging. In recent study, transfer learning techniques have been proven useful to apply knowledge from a large dataset to a different smaller set of target data [4].

In this paper, we attempt to tackle the problem of training the deep neural networks for fine-grained image classification with limited training data. We take advantage of a few models pre-trained on ImageNet [5], modify the last few fully connected layers and fine-tune the models on Oxford 102 Flower dataset. We also apply some visualization techniques to better understand and interpret the trained model, including activation maximization, saliency map [6] and grad-CAM [7].

The rest of the paper is organized as following: section 2 will introduce the related work, section 3 will cover the visualization techniques used in our work, section 4 will walk through the experiments and section 5 will conclude this article.

2. Related Work

There are several noticeable works done on fine-grained classification and transfer learning.

Fine-grained classification. In [8] Maria *et al.* introduced the dataset known as VGG 102 Flower and investigated on multiple kernel SVM for multiclass flower images classification. A few image features are manually picked and combined to describe the image. Some other work focused on automatic taxonomic classification using CNN [9, 10, 11] and keep achieving better result each year.

Feature Extraction and Fine-tuning. Off-the-shelf CNN features [9, 12] has shown its power in a plenty of recognition problems. Since the release of benchmark datasets like ImageNet [5] and Places [13], there are increasing number of researchers pre-training their model before applying on other vision problems. While there is always gap between features the pre-trained model learned and the features in target dataset. As a result, fine-tuning [14] has become an essential technique to solve this problem.

Transfer Learning. It's a common assumption in

machine learning algorithms that training data and testing data must conform the same statistical distribution. Unfortunately, it is not always the case. Some data of interest is limited or out of access, while one only has sufficient training data in another domain of interest. Since the concept of transfer learning introduced in mid 90s [15], there come the research in multiple task learning [16] and domain adaptation [17, 18, 19]. In some recent approach, Tzeng *et al*. [20] achieved feature adaptation across domain and tasks using a shared CNN, Hong *et al*. [21] transferred rich semantic information from source categories to target categories using attention model.

3. Visualization of Deep Network

CNN has long been known as "black boxes" and been argued as lack of interpretability, because it's hard to understand exactly, which function contribute to the classification and what the model has learned during tremendous iteration of training.

To tackle this problem, we attempt to explain the classification model using visualization. There are a few visualization techniques used in this paper.

3.1. Saliency Maps

Suppose in a CNN based image classification model, we want to know which exact feature or property that makes the input image classified into certain class. This visualization could help us deeper understand how the model capture the image and how it makes the decision.

The concept of saliency maps was first proposed by Simonyan *et al.* [23]. The idea is pretty straightforward. Given a classification model, an input image and a class of interest, the saliency map tells how the image spatial information support a particular class in a given image. It is computed as the gradient of output category score with respect to the input image.

$$\omega = \frac{\partial S_c}{\partial \mathrm{I}}\bigg|_{I_0}$$

This expression also indicates which pixels change lead to the most class score change. In such way, we can draw the saliency map using these gradients to highlight the input regions that cause the most change toward the output class score.

Concretely, the saliency map is computed as follows. Given a class c and image I_0, we first compute the derivative using back-propagation. Next, we rearrange the elements of the vector to form the dimension of the original image.

In this way, the saliency map could be generated using a trained classification CNN with one pass of back-propagation, no additional information or computation is required.

3.2. Class Activation Maximization

Another techniques that is recently proposed in [23] is class model visualization. This will generate an image that is most representative to a given class, according to the classification model.

$$argmax_I \, S_c(I) - \lambda \, \|I\|_2^2$$

This is achieved by maximizing the class activation by iteratively back-propagate the gradient and optimize the image. Different from the saliency map, the computation of class model visualization is more expensive and allow an image to be generated.

3.3. Gradient-weighted Class Activation Map

Gradient-weighted Class Activation Map, known as Grad-CAM[7], is another method to visualize the attention of model over input. It's applicable to a variety of CNN based models and tasks, including VGG-like CNNs with fully connected layer, CNN used for structured output and CNN that tackle with VQA with multi-modal inputs.

Unlike the approaches in 3.1 and 3.2, in classification task, grad-CAM uses the gradient of the target class and utilize the last convolutional layer to acquire spatial information flowing into the fully-connected layer.

Specifically, we first compute the gradient of the class c, (denoted as y^c), with respect to feature maps A^k of a convolutional layer, denoted as $\frac{\partial y^c}{\partial A^k}$. The following equations denotes the neuron importance weights.

$$\alpha_k^c = \overbrace{\frac{1}{Z}\sum_i \sum_j}^{\text{global average pooling}} \underbrace{\frac{\partial y^c}{\partial A_{ij}^k}}_{\text{gradients via backprop}}$$

In this manner, we can compute the weighted combination of forward activation maps and follow it by a ReLU to get the grad-CAM result.

$$L_{\text{Grad-CAM}}^c = ReLU \underbrace{\left(\sum_k \alpha_k^c A^k \right)}_{\text{linear combination}}$$

4. Experiments and Analysis

4.1. Dataset

The Oxford 102 Flower dataset is a well-known subcategory recognition dataset proposed by Nilsback *et al.* [22]. The dataset contains 102 species of flowers, with total count of 8189 images. The number of images in each category varies from 40 to 250. In this work, we follow the official protocol on dataset split, where training and validation data each contains 1020 images (10 for each class), and the rest 6149 as test data.

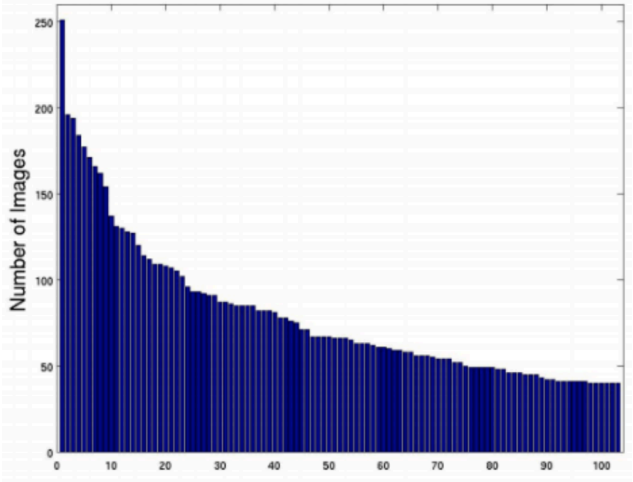


Figure 4.1. The distribution of sample size in each class

4.2. System Setup

Our experiments are based on Python 2.7.12 on Ubuntu 16.04.4 LTS with a single GPU support GeForce GTX 1070 Ti. The model architecture is built in keras 2.1.5, using tensorflow 1.4.0 as backend.

This code on [25] has also been tested on Mac OS 10.13.4.

4.3. Procedure

In our experiment, we first mount the dataset according to the official protocol and start with a simple convolutional neural network so that we can get some quick baseline result that can guide the later experiment.

The architecture of the baseline model is shown in figure 4.2. A few experiments are conducted in this phase:

(a). Naively train the model with training set with size 1020.

(b). Train the model with training set using data augmentation.

(c). Train the model using more 6149 images as training set.

The data augmentation is achieved using keras ImageDataGenerator class, including rotation, shear, shift, zoom, flip and zca whitening. For implementation details, please refer to the GitHub repository [25].

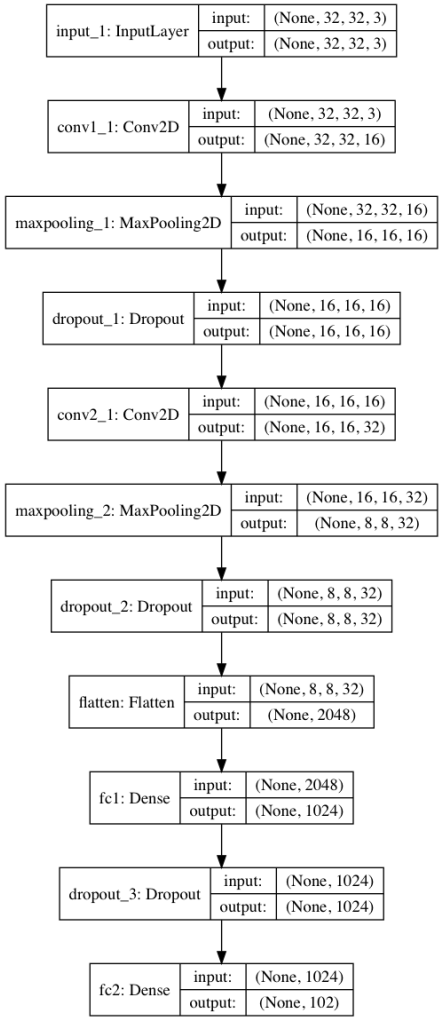
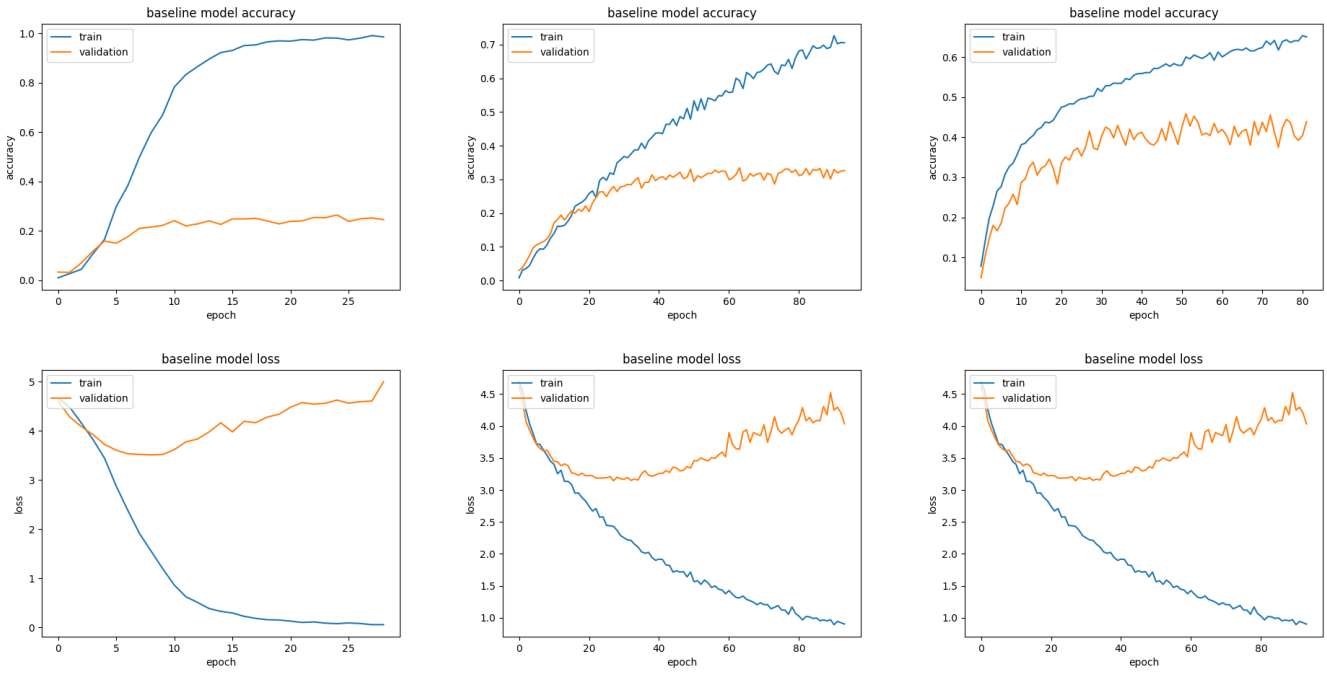


Figure 4.2. the architecture of a baseline model

We then incorporate the architecture and weights of some pre-trained CNN, including VGG16, VGG19 [24] and inception-v3 [25]. All the pre-trained model weights are from ImageNet, obtained from keras library. We retrieve the weights without the top softmax layer. For VGG16 and VGG19, we add one fully-connected layer before the softmax prediction layer. For inception-v3, the softmax layer is directly built on top of the average pooling layer. All models have included dropout to avoid overfitting.

We take a two-step approach in training. First, we train the fully-connected layer only, which are the one or two layers on the top, using normal learning rate, to build a weak classifier. Then we allow the lower layers to be trainable with much smaller learning rate, so that the learned features will not be contaminated too much.



(a) (b) (c)

Figure 4.3 (a~c) Learning curves of Baseline model
(a) naïve training on 1020 images
(b) training on 1020 images with data augmentation
(c) training on 6149 images with data augmentation

In the phase of top-layer training, the number of epochs is set to be small to avoid overfitting.

The whole training process typically completes in 2 hours on a single GPU in our experiment, with softmax as loss function and Adam or SGD as optimizer.

4.4. Results and Analysis

The results of the baseline model are shown in Table 4.3 and learning curves shown in figure 4.1.

	top-1 accuracy
a)	0.181
b)	0.302
c)	0.448

Table 4.1. baseline model experiment result
(a) naïve training on 1020 images
(b) training on 1020 images with data augmentation
(c) training on 6149 images with data augmentation

We can see that the baseline model greatly suffers from over fitting in setting (a), while it gets significantly better with data augmentation and increasing the size of training samples.

The comparison between a) and b) is a great indication that data augmentation offers a great boost in performance especially in small sample training problem. As a result, in the rest of the paper, we conduct data augmentation whenever possible.

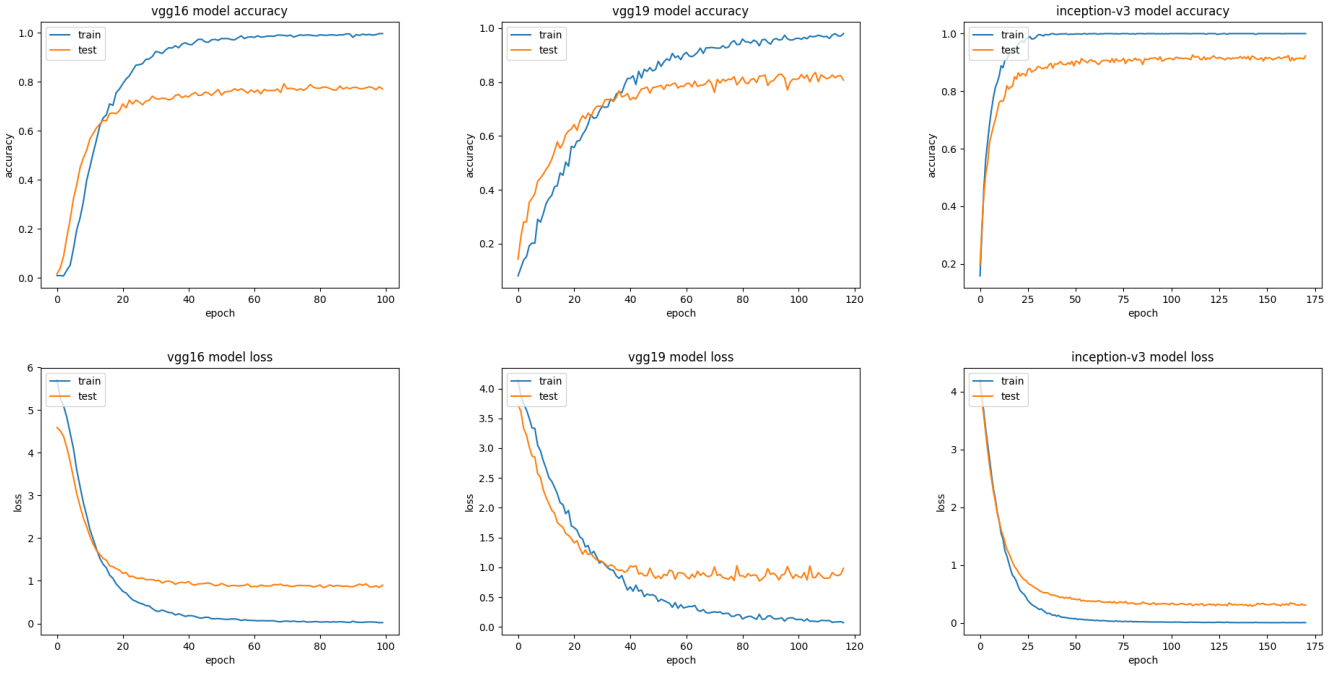
The test results of the fine-tuning experiments are shown in Table 4.2. and learning curves shown in figure 4.4.

	VGG16	VGG19	Inception-v3
a)	0.757	0.781	0.904
b)	0.889	0.925	0.962

Table 4.2. fine-tuning models, results measured by mean class top-1 accuracy
a) training on 1020 official training set
b) training on 6149 images

Borrowing the learned features from ImageNet pre-trained models, the performance has greatly improved. It has shown the potential of transfer learning in fine-grained classification task with extremely small sample data.

So far, we have discussed the performance of classifiers but still cannot interpret the model so well. We want to explore the questions like, how the



(a) (b) (c)

Figure 4.4. (a-c) Learning curves of Fine-tuning models
(a) VGG16
(b) VGG19
(c) Inception-v3

Top-1 Accuracy	Methods	Year
76.3%	SVM + kNN [8]	2009
81.4%	SVM + bag-of-words [26]	2011
86.8%	CNN + SVM [9]	2014
94.5%	Bayesian [27]	2016
95.8%	CNN with Selective Joint Fine-tuning [28]	2017
90.4%	Ours	2018

Table 4.3. Comparison between our work and others

classification is made, what specific features are learned and what makes one category distinguishable from the others. To cope with these problems, we take a deeper look inside the models using a few visualizations tools described earlier in this paper.

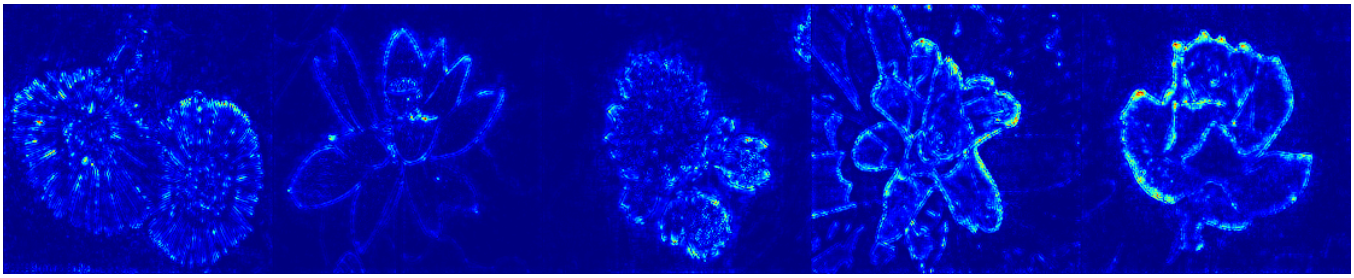
Figure 4.5 shows the visualization results on a few example test images using trained VGG16 model. This serves as a great indication of how the model perceive the visual input. We can see from figure 4.6 that the border of the flower petal, in which area has high spatial frequency, usually have high saliency response. From grad-CAM results we can also see that the model has a focus of the flower rather than its surroundings. Furthermore, we could utilize this method to conduct unsupervised localization tasks from a trained classification model.

Table 4.3 shows a few results by other groups using Oxford 102 Flower dataset.

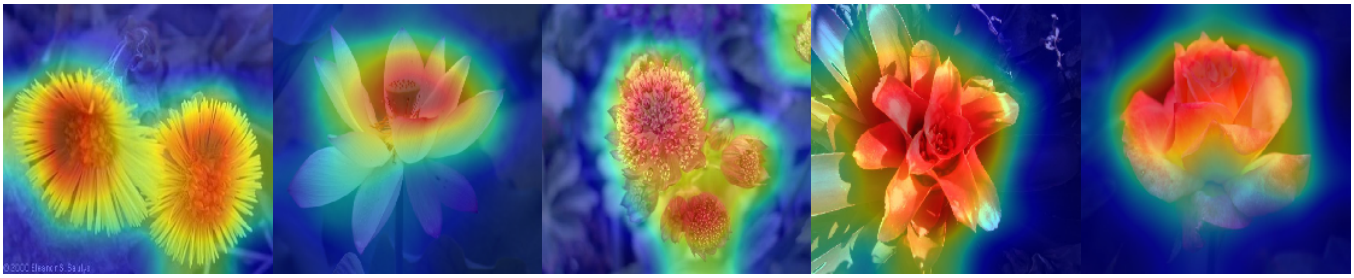
Our result does not beat the current state-of-art, but we achieve relative good model performance with limited computational resources and without outside data. More advanced methods require longer time training time and may not be available on single GPU.

(i) Original images

(ii) Saliency

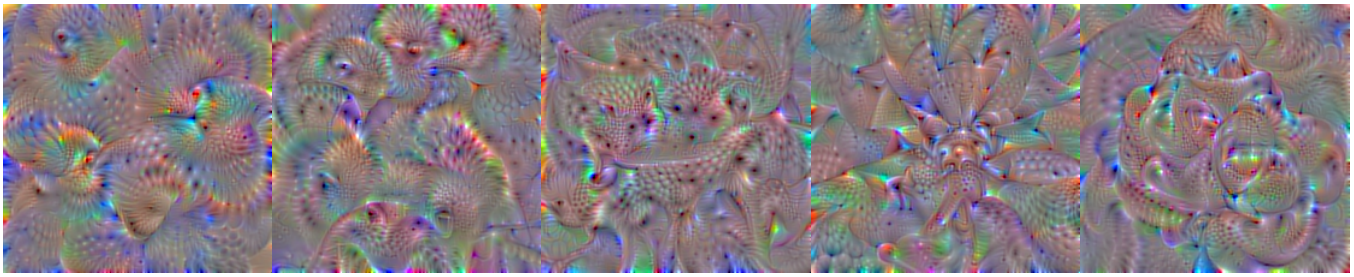


(iii) Grad-CAM



(a) (b) (c) (d) (e)

Figure 4.5 (i), (ii), (iii). Visualization of test examples
(a) 5: Tiger lily
(b) 7: Lotus
(c) 33: Mexican aster
(d) 100: Trumpet creeper
(e) 73: Rose



(a) (b) (c) (d) (e)

Figure 4.6. Class Activation Maximization
(a) 5: Tiger lily
(b) 7: Lotus
(c) 33: Mexican aster
(d) 100: Trumpet creeper
(e) 73: Rose

5. Conclusion and Future Work

In this paper, we conducted transfer learning on a fine-grained classification program and achieved a 90.4% top-1 accuracy on Oxford 102 Flower dataset with no outside data. We also utilize a few visualization techniques to further validate and understand the trained convolutional neural network. The visualization results indicate that the model captures the visual property of flower rather than irrelevant background. The generated image is also of great value in aesthetic on how machine perceive the world.

There are a few directions to further the study of this paper. First, the availability of outside data could potentially boost the performance of current model. Though they probably are of different distribution, it could help if we can bridge a connection between the relative scarce target dataset and abundant source dataset. Second, utilize more image preprocessing. Since the classification is primarily made on the region of flower rather than the whole image, we could conduct unsupervised segmentation as an image preprocessing step before the model training. With the elimination of irrelevant information, we expect an improve on the performance. Last but not the least, taking the domain knowledge of botany. The recognition usually requires analysis of the composition of the flower, which may include the carpel, stamen and lodicule. This is a great indication that a part-based ensemble model could help in this scenario. A part-based model has to be implemented with fine region proposal.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems, 2012.

[2] He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[3] Schmidhuber, Jürgen. "Deep learning in neural networks: An overview." *Neural networks* 61 (2015): 85-117.

[4] Pan, Sinno Jialin, and Qiang Yang. "A survey on transfer learning." *IEEE Transactions on knowledge and data engineering* 22.10 (2010): 1345-1359.

[5] Russakovsky, Olga, et al. "Imagenet large scale visual recognition challenge." *International Journal of Computer Vision* 115.3 (2015): 211-252.

[6] Simonyan, Karen, Andrea Vedaldi, and Andrew Zisserman. "Deep inside convolutional networks: Visualising image classification models and saliency maps." *arXiv preprint arXiv:1312.6034* (2013).

[7] Selvaraju, Ramprasaath R., et al. "Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization." *arXiv preprint arXiv:1610.02391* (2016).

[8] M-E. Nilsback. An Automatic Visual Flora – Segmentation and Classification of Flowers Images. PhD thesis, University of Oxford, 2009.

[9] Razavian, Ali Sharif, et al. "CNN features off-the-shelf: an astounding baseline for recognition." Computer Vision and Pattern Recognition Workshops (CVPRW), 2014 IEEE Conference on. IEEE, 2014.

[10] Liu, Yuanyuan, et al. "Flower classification via convolutional neural network." Functional-Structural Plant Growth Modeling, Simulation, Visualization and Applications (FSPMA), International Conference on. IEEE, 2016.

[11] Wang, Yu-Xiong, and Martial Hebert. "Learning to learn: Model regression networks for easy small sample learning." *European Conference on Computer Vision*. Springer, Cham, 2016.

[12] Donahue, Jeff, et al. "Decaf: A deep convolutional activation feature for generic visual recognition." *International conference on machine learning*. 2014.

[13] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. "Learning Deep Features for Scene Recognition using Places Database." Advances in Neural Information Processing Systems 27 (NIPS), 2014.

[14] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 580–587, 2014.

[15] http://socrates.acadiau.ca/courses/comp/dsilver/NIPS95 LTL/transfer.workshop.1995.html

[16] R. Caruana, "Multitask learning," Machine Learning, vol. 28(1), pp. 41– 75, 1997

[17] L. Duan, D. Xu, I. W.-H. Tsang, and J. Luo. Visual event recognition in videos by learning from web data. IEEE Transactions on Pattern Analysis and Machine Intelligence, 34(9):1667–1680, 2012.

[18] J. Huang, A. Gretton, K. M. Borgwardt, B. Scholkopf, and ¨ A. J. Smola. Correcting sample selection bias by unlabeled data. In Advances in neural information processing systems, pages 601–608, 2006.

[19] M. Long and J. Wang. Learning transferable features with deep adaptation networks. CoRR, abs/1502.02791, 1:2, 2015.

[20] E. Tzeng, J. Hoffman, T. Darrell, and K. Saenko. Simultaneous deep transfer across domains and tasks. In Proceedings of the IEEE International Conference on Computer Vision, pages 4068–4076, 2015.

[21] S. Hong, J. Oh, B. Han, and H. Lee. Learning transferrable knowledge for semantic segmentation with deep convolutional neural network. In IEEE Conference on Computer Vision and Pattern Recognition, 2016.

[22] M.-E. Nilsback and A. Zisserman. Automated flower classification over a large number of classes. In Computer Vision, Graphics & Image Processing, 2008. ICVGIP'08. Sixth Indian Conference on, pages 722–729. IEEE, 2008.

[23] Simonyan, Karen, Andrea Vedaldi, and Andrew Zisserman. "Deep inside convolutional networks:

Visualising image classification models and saliency maps." *arXiv preprint arXiv:1312.6034* (2013).

[24] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." *arXiv preprint arXiv:1409.1556* (2014).

[25] Szegedy, Christian, et al. "Rethinking the inception architecture for computer vision." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016.

[26] Chai, Yuning. "Recognition between a large number of flower species." *University of Oxford* (2011).

[27] Kim, Yong-Deok, et al. "Learning to select pre-trained deep representations with bayesian evidence framework." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016.

[28] Ge, Weifeng, and Yizhou Yu. "Borrowing treasures from the wealthy: Deep transfer learning through selective joint fine-tuning." *Proc. IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI*. Vol. 6. 2017.