Improving Fine-grained Image Classification by Edge Detection

Li Ding
Goergen Institute for Data Science
University of Rochester
Rochester, NY 14627
lding7@ur.rochester.edu

Abstract

Suppose given two images, one for fried calamari and another for onion rings, how can an image classification model distinguish them as they look quite similar even to human? Such fine-grained image classification problems are still very challenging while general image classification models have achieved impressive success within the last few years. In this paper, we come up with the idea as to look at the image in a different way. In order to observe subtle differences of objects in their shapes, which may be overridden by their similarity in colors, we use edge detection techniques to focus on the edges and boundaries of images. We propose two merging CNNs structures based on edge detectors and deep CNNs as ways to combine them together. Experimental results on the Food-101 dataset demonstrate that the proposed new structures outperform the popular deep CNNs model on this task. In addition, the proposed structures can be treated as a very flexible plug-in that can be applied to different models on different tasks. Our work is paralleled with most of other work on the topic of fine-grained image classification.

1. Introduction

Fine-grained image recognition is very challenging because it needs to discern subtle difference between classes while dealing with the scarcity and noise of training data at the same time. It usually concerns the task of distinguishing the sub-ordinate categories of some base classes such as birds [1, 2], flowers [3, 4], cars [5, 6, 7], food [8, 9, 10, 11, 12], etc. It is different from the general classification problems [13] in that the differences among object classes are more subtle, and thus it is more difficult to distinguish them. Even though impressive success has been achieved by general image classification within the last few years by some deep CNNs models [14, 28, 29, 15, 16], it is still very challenging to recognize object classes with ultra-fine granularity.

Figure 1. Example of Food Classification problem. These three images are very hard to distinguish even for human.

1.1. Problem

Take food classification as an instance: as shown in Figure 1, how can we distinguish fried calamari and onion rings? In addition, what if there is another image of onion rings but taken in a dark environment? Such problems cannot be solved with the popular general image classification methods which are designed for and trained on large scale images with some obvious differences, especially in color.

But if we take a careful look at the image, it can been observed that there are still some subtle differences between fried calamari and onion rings. Comparing to fried calamari, onion rings tend to have a smoother surface which is to say, the circle of onion rings is smoother. But such subtle difference is very likely to be overridden by their similarity in color.

1.2. Idea

To solve the problem discussed above, we propose a novel approach using edge detection methods to improve the modern structure of deep neural networks. Just as the example discussed above, our intuition is to treat edge as a feature, which is to make edge as an additional input to the neural networks. Specifically, we transform every original image into a corresponding edge image, which contains all the edges and boundaries as the result of the edge detection process. As shown in Figure 2, the example images for fried

Figure 2. Example of edge detection. The edge images are focusing on boundaries and shape of the object in the images.

calamari and onion rings are transformed into edge images by some edge detector. Now we can use the edge images as another input for some neural network structure and boost the accuracy of a fine-grained classification model. Though the edge image is also obtained from the original image, it is obvious to see the difference between it and the original image, and thus additional information can be extracted for fine-grained image classification tasks.

The major contributions of our work can be concluded as follows:

- We introduce a novel idea of using edge detection methods along with modern deep learning structures.
- Our experiments prove that by edge detection, additional information can be obtained to improve the performance of neural networks.
- We propose two structures combining edge detection methods and deep neural networks for the fine-grained image classification problems. Experiment results show that the accuracy is improved by new structures.

2. Related Work

Some previous work has been done in research of the following topics.

2.1. Fine-grained Image Classification

Fine-grained image classification needs to discern subtle differences among similar classes. The majority of existing approaches have thus been focusing on localizing and describing discriminative object parts in fine-grained domains. Various pose-normalization pooling strategies combined with 2D or 3D geometry have been proposed for recognizing birds [1, 2]. The main drawback of these approaches is that part annotations are significantly more challenging to collect than image labels. Instead, a variety of methods have been developed towards the goal of finding object parts in an unsupervised or semi-supervised fashion. Krause et al. [17] combined alignment with co-segmentation to generate parts without annotations. Zhou and Lin [19] proposed a structure to improve CNNs by exploring bipartite-graph labels. Lin et al. [18] proposed an architecture that uses two separate CNNs feature extractors to model the appearance according to where the parts are and what the parts look like.

2.2. Edge Detection

Edge detection is a basic image processing technique for finding the boundaries of objects within images. It usually works by detecting discontinuities in brightness. There are many famous edge detectors like Canny detector [20], Sobel-Feldman operator [21], Roberts cross [22], Scharr operator [23] and Prewitt operator [24]. Shrivakshan, G. T. [26] did a comparison of various edge detection techniques used in image processing. Besides, there are also some modern techniques to detect edge and boundaries. P. Arbelaez [25] proposed a boundary extraction method using Ultrametric Contour Maps. Dollr, P., & Zitnick, C. L. [27] used structured forest for fast edge detection.

3. Methodology

In this section, we describe the mathematical basis of our idea in detail, as well as our proposed framework of using edge detection along with deep convolutional neural networks, which is compatible to most popular architectures like AlexNet [14], GoogLeNet [28], VGGNet [29] and ResNet [15]. Briefly, we add edge image as input and train another neural-network structure, then try different merging methods to combine it with the original networks with input of original images. The resulting structure can be optimized by a global back-propagation.

3.1. Deep Convolutional Neural Networks

Suppose that we are given a set of n images $X = \{(x, y), \cdots\}$ for training, where each image x is annotated with one of k fine-grained labels as $y = 1, \cdots, k$.

Let $x \in \mathbb{R}^d$ denote the input feature of the last fully-connected layer, which generates k scores $f \in \mathbb{R}^k$ through a linear function $f = W^T x$ defined by the parameters $W \in \mathbb{R}^{d \times k}$. In a sense, the last layer of neural networks is to minimize the negative log-likelihood over the training data, which is,

$$\min_W \sum_{(x,y \in X)} -\log P(y|x, W) \tag{1}$$

where the softmax score,

$$P(i|x, W) = \frac{e^{f_i}}{\sum_{j=1}^{k} e^{f_i}} = p_i \tag{2}$$

encodes the posterior probability of image x being classified as the ith fine-grained class. [19]

3.2. Edge Detectors

In general, edge operators are discrete differentiation operators, computing an approximation of the gradient of the

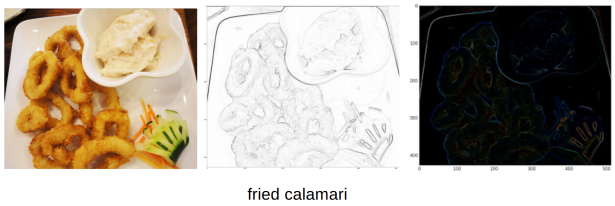

Figure 3. Example of edge detection using Sobel-Feldman operator. Left is the original image, middle for the gray-scale edge image and the right one for the 3D (RGB) edge image.

image intensity function. For instance, Sobel-Feldman operator [21] uses two 3×3 kernels which are convolved with the original image to calculate approximations of the derivatives - one for horizontal changes, and one for vertical. If we define A as the source image, and G_x and G_y are two images which at each point contain the horizontal and vertical derivative approximations respectively, the computations are as follows:

$$G_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} * A, G_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * A \tag{3}$$

where $*$ here denotes the 2-dimensional signal processing convolution operation.

In order to extract most information from the image, we use the detectors separately on the three dimension (RGB) of the original image. Therefore, we get another edge image with the same size as the original image. An example of the results of edge detection is shown in Figure 3.

We also compare different edge detection operators. An example of such comparison is shown in Figure 4. Due to the limitation of time, we choose the famous Sobel-Feldman operator for implementation in this work.

3.3. Merged Structures

3.3.1 Merge-at-last

We propose a merge-at-last framework for adding edge image into neural networks.

Let $x^{(1)} \in \mathbb{R}^d$ denote the input feature of the last fully-connected layer, which is given by some networks with the input of original images. $x^{(2)} \in \mathbb{R}^d$ denote the input feature of the last fully-connected layer, which is given by some networks with the input of edge images.

We concatenate $x^{(1)} \in \mathbb{R}^d$ and $x^{(2)} \in \mathbb{R}^d$ to form $x \in \mathbb{R}^{2 \times d}$ as the input feature of the new last fully-connected layer. Then the softmax process is as introduced in Section 3.1.

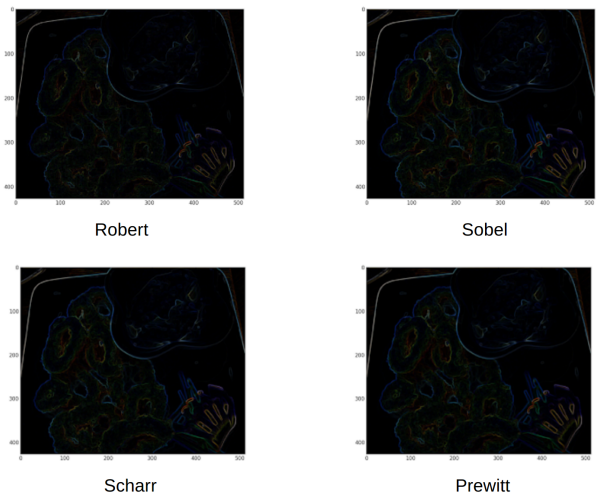

Figure 4. Example of edge detection using Different operators (Sobel-Feldman operator [21], Roberts cross [22], Scharr operator [23], Prewitt operator [24]).

3.3.2 Merge-at-first

We propose another merge-at-first framework for adding edge image into neural networks.

Suppose the input images are scaled in 256×256 with 3 dimension (RGB). Let $x^{(1)} \in \mathbb{R}^{256 \times 256 \times 3}$ denote the original images. $x^{(2)} \in \mathbb{R}^{256 \times 256 \times 3}$ denote the edge images.

We concatenate $x^{(1)}$ and $x^{(2)}$ to form $x \in \mathbb{R}^{256 \times 256 \times 6}$ as the input feature of the new input layer of some neural networks. Then the training and softmax process are again as introduced in Section 3.1.

4. Experiment and Result

In this section, the results of all the experiments are shown. Since this is an exploratory work, we simply use part of the Food-101 data [30] to ensure the efficiency of the experiment. Specifically, we choose the first ten classes to test our idea and approaches.

4.1. Implementation

We implement four different models on the same data. The experiments are all done on one laptop with *Nvidia* GTX-1070 GPU, using *Keras* library with *Tensorflow-Cudnn* back-end. The objective function takes the categorical cross-entropy as the global loss and we use mini-batch stochastic gradient descent to optimize the models.

- **VGG-16 (Baseline)**

  At first, we use the popular VGG-16 structure as our baseline model. Weights of the convolutional layers

Methods	Training Accuracy	Testing Accuracy
VGG-16 (Baseline)	92.08%	79.63%
VGG-16 (Edge Image)	**99.59%**	58.28%
Merge-at-last	92.21%	**81.47%**
Merge-at-first	83.76%	58.44%

Table 1. Results of Experiments

are pre-trained on ImageNet. We fine-tune the fully-connected layers and configure the output layer for our task.

- **VGG-16 with edge images only**

  We use Sobel Edge Detector to transform the original images into edge images, and use it to replace the original input of the VGG-16 Network. Similarly then, weights of the convolutional layers are pre-trained on ImageNet. We fine-tune the fully-connected layers and configure the output layer for our task.

- **Merge-at-last Structure**

  We combine the above two models by concatenating their input of the two last fully connected layers, then train a new fully connected layer with soft-max activation for classification. Weights of the convolutional layers are also pre-trained on ImageNet.

- **Merge-at-first Structure**

  We combine the original images and their corresponding edge images to make a 6-dimension input (as introduced in section 3.3.2) for the VGG-16 Structure. Then we retrain the first convolutional layer and the last fully connected layers. Weights of other convolutional layers are pre-trained on ImageNet.

4.2. Results

The results of the above four experiments are shown in table 1. Both training accuracy and testing accuracy are selected among all the epochs before the models get overfitting.

4.2.1 Edge detection makes a difference

Comparing the results of VGG-16 (Baseline) and VGG-16 (Edge Image), we can see that if we replace the input with edge images, the same VGG model gets a higher training accuracy but a much lower testing accuracy. So first we can conclude that edge images are different from original images for this fine-grained classification task. Though edge images are also obtained from original images, they are giving different information to the Deep CNNs model.

More specifically, we may observe that the edge images can perform a nearly 100% training accuracy, because they focus on the inner edge and shape of the object, which are more distinguishable. This partly proves our assumption in the beginning of this paper. But the testing accuracy is much lower, perhaps it's because the VGG is pretrained on ImageNet, which is not that proper to use directly on edge images.

4.2.2 Merging makes a difference

In this work, we tried two different structures for adding edge information into the Deep CNNs structure, Merge-at-last structure and Merge-at-first structure. As we can see in Table 1, the Merge-at-last model performs better than the well-trained VGG-16 model, which means that it successfully uses some additional information from edge detection to improve the classification accuracy. The Merge-at-first model does not perform well, mainly because it requires to retrain the whole CNNs with those convolutional layers. Since this model is not trained on large scale image dataset (e.g. ImageNet), such comparison is not fair. However, we can still conclude that edge detection makes a difference and can benefit fine-grained image classification. The idea is worth trying and we are looking forward to other future work on it.

5. Conclusion

This paper first proved that even popular deep CNNs cannot well learn the shape of the object, which may be provided by edge detection, in fine-grained image classification tasks. Then we proposed two different merging structures combining edge detection with modern deep CNNs. Both of the structures are flexible and can be applied to different edge detectors as well as different deep CNNs. Besides, both resulting structures can be optimized by a global back-propagation.

Since the proposed idea is quite general, there are several future directions to our work. First of all, more experiments on larger data sets can be done to make the conclusion of this work more convincing. Secondly, in order to make sure how much improvement edge detection can give to fine-grained classification tasks, the two proposed structures can be implemented with different edge detectors and different CNNs. Moreover, this is just a first trial. We hope to see other work on the same idea that using 'old-fashioned' edge detectors with modern computer vision models like CNNs, CRF to get an improvement.

Acknowledgment

The authors would like to thank Prof. Chenliang Xu for his valuable advice on this work.

References

[1] T. Berg, J. Liu, S. W. Lee, M. L. Alexander, D. W. Jacobs, and P. N. Belhumeur. Birdsnap: Large-scale fine-grained visual categorization of birds. In CVPR, 2014.

[2] S. Branson, G. V. Horn, C. Wah, P. Perona, and S. Belongie. The ignorant led by the blind: A hybrid human-machine vision system for fine-grained categorization. Int. J. Comput. Vis., 108(1-2):329, 2014.

[3] A. Angelova and S. Zhu. Efficient object detection and segmentation for fine-grained recognition. In CVPR, 2013.

[4] M. Nilsback and A. Zisserman. Automated flower classification over a large number of classes. In ICVGIP, 2008.

[5] J. Krause, M. Stark, J. Deng, and L. Fei-Fei. 3D object representations for fine-grained categorization. In ICCV Workshop on 3D Representation and Recognition, 2013.

[6] Y. Lin, V. I. Morariu, W. H. Hsu, and L. S. Davis. Jointly optimizing 3D model fitting and fine-grained classification. In ECCV, 2014.

[7] M. Stark, J. Krause, B. Pepik, D. Meger, J. J. Little, B. Schiele, and D. Koller. Fine-grained categorization for 3D scene understanding. In BMVC, 2012.

[8] O. Beijbom, N. Joshi, D. Morris, S. Saponas, and S. Khullar. Menu-Match: Restaurant-specific food logging from images. In WACV, 2015.

[9] L. Bossard, M. Guillaumin, and L. V. Gool. Food-101 - mining discriminative components with random forests. In ECCV, 2014.

[10] A. Myers, N. Johnston, V. Rathod, A. Korattikara, A. Gorban, N. Silberman, S. Guadarrama, G. Papandreou, J. Huang, and K. Murphy. Im2Calories: towards an automated mobile vision food diary. In ICCV, 2015.

[11] S. Yang, M. Chen, D. Pomerleau, and R. Sukthankar. Food recognition using statistics of pairwise local features. In CVPR, 2010.

[12] Zhou, F., & Lin, Y. Fine-grained Image Classification by Exploring Bipartite-Graph Labels. In CVPR, 2016.

[13] M. Everingham, S. M. A. Eslami, L. V. Gool, C. K. I. Williams, J. M. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. Int. J. Comput. Vis., 111(1):98136, 2015.

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In NIPS, 2012.

[15] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rec- tifiers: Surpassing human-level performance on ImageNet classification. CoRR, abs/1502.01852, 2015.

[16] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. CoRR, abs/1502.03167, 2015.

[17] J. Krause, H. Jin, J. Yang, and L. Fei-Fei. Fine-grained recognition without part annotations. In CVPR, 2015.

[18] T.-Y. Lin, A. R. Chowdhury, and S. Maji. Bilinear CNN models for fine-grained visual recognition. In ICCV, 2015.

[19] F. Zhou and Y. Lin. Fine-grained Image Classification by Exploring Bipartite-Graph Labels. In CVPR, 2016.

[20] J. Canny (1986) "A computational approach to edge detection", IEEE Trans. Pattern Analysis and Machine Intelligence, vol 8, pages 679714.

[21] Sobel, I., Feldman, G., "A 3x3 Isotropic Gradient Operator for Image Processing", presented at the Stanford Artificial Intelligence Project (SAIL) in 1968.

[22] Roberts, Lawrence Gilman. Machine perception of three-dimensional soups. Diss. Massachusetts Institute of Technology, 1963.

[23] Scharr, Hanno, 2000, Dissertation (in German), Optimal Operators in Digital Image Processing.

[24] J.M.S. Prewitt "Object Enhancement and Extraction" in "Picture processing and Psychopictorics", Academic Press,1970

[25] P. Arbelaez. Boundary Extraction in Natural Images Using Ultrametric Contour Maps. Proceedings 5th IEEE Workshop on Perceptual Organization in Computer Vision (POCV'06). 2006.

[26] Shrivakshan, G. T., & Chandrasekar, C. (2012). A comparison of various edge detection techniques used in image processing. IJCSI International Journal of Computer Science Issues, 9(5), 272-276.

[27] Dollr, P., & Zitnick, C. L. (2013). Structured forests for fast edge detection. In Proceedings of the IEEE International Conference on Computer Vision (pp. 1841-1848).

[28] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. CoRR, abs/1409.4842, 2014.

[29] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. CoRR, abs/1409.1556, 2014.

[30] Bossard, Lukas, Matthieu Guillaumin, and Luc Van Gool. "Food-101mining discriminative components with random forests." European Conference on Computer Vision. Springer International Publishing, 2014.