CS230

CS 230: Fashion Product Recognition in Fine-Grained Visual Categorization

Renato Baba
Graduate School of Business
Stanford University
rbaba@stanford.edu

Klemen Cas
Graduate School of Business
Stanford University
klemen.cas@stanford.edu

Yehia Khoja
Graduate School of Business
Stanford University
ykhoja@stanford.edu

Abstract

In this work, we analyze the problem of multi-label classification in Fine-Grained Visual Categorization (FGVC) in the fashion industry. This is a class of problems that is considered difficult to address with regular computer vision algorithms because of the nuanced difference between classes. For example, differentiating between a white and a cream-colored sweater might prove difficult for a standard model. Hence, we employed transfer learning techniques to fine-tune a pretrained model to identify these nuanced differences. We have used a dataset of 1,014,544 images with their respective labels provided by a Kaggle competition addressing this problem. Furthermore, we proceed to visualize the learned parameters and activations of the trained model to understand how it was able to categorize these fine-grained images. We used accuracy, defined as the number of correct predictions for each image, as our main metric for training the model. In addition, we use the F1-score as a secondary metric. In the end, our fine-tuned model was able to achieve 97% accuracy on the test set, and 45% F1-score.

1 Introduction

As online shopping continues to grow, it is becoming increasingly important to automate product recognition and labeling. For example, a customer can search for a product more easily, and a seller can tag and label images containing the product quickly. However, automatic fashion product recognition remains a tough challenge for various reasons. First, pictures of the same product might be taken at different angles, lighting, backgrounds, and levels of occlusion. Second, fashion products in general have fine-grained categories, where the differences might be subtle.

In this work, we use a dataset consisting of images and respective ground truth labels representing 228 possible categories. Each image can have one or more labels as the ground truth. For example, the labels might indicate the type of product (shirt, pants, shoes, etc), the color, material (wool, leather, nylon, etc), or other features. Therefore, this is a multi-label classification problem. We then proceed to using transfer learning with an Inception v3 model (Szegedy et al., 2015) trained on ImageNet to output the predicted categories of a product image. Further, we present visualizations of the learned parameters and their interpretation.

2 Related work

The issue of performing object detection with invariance in pose and lighting is an issue that's been studied by LeCun et al. (2004), which was before the rise of Convolutional Neural Networks (CNNs). Once CNNs became more popular, there were several more detailed approaches like the work of Qian et al. (2014) used a method that generates embeddings to create a sense of distance between image categories.

Nevertheless, our team decided to take a different approach, which is similar to standard image recognition problem. Namely, we used transfer learning to fine-tune the Inception v3 network (Szegedy et al., 2015) using our data set. In addition, we have found work done by Bartyzal (2017) to train the Inception v3 network for multi-label problems. In addition, we found previous work from the previous iteration of the Kaggle competition as described by Dziedzic & Miziuła (2017).

Finally, after training the model we have used the work by Zeiler & Fergus (2013) to visualize the learning of our network in order to illustrate the features that it learned to look for during training.

3 Dataset and Features

3.1 Dataset Description

Our dataset contains 1,014,544 images supplied by the organizers of the contest *iMaterialist Challenge (Fashion) at FGVC5* on Kaggle. Each image can have multiple ground truth labels out of 228 possible categories. The labels might indicate the type of product (shirt, pants, shoes, etc), the color, material (wool, leather, nylon, etc), or other features. Figure 1 shows a sample image from the training dataset.

Figure 1: Sample training image with ground truth labels

The dataset was packaged in a `.json` file which contained

- image ID number
- image URL
- ground truth labels

We have used a python script to download the images. Upon analyzing the dataset, we discovered that we have imbalances in the labels. The discussion lead by Sban (2018) on the Kaggle competition discussions highlighted key findings in the data. Figure 2 shows the frequency of each label in the complete dataset using a histogram with labels on the horizontal access, and count of occurrences on the vertical access. Moreover, we look at the top 10 most frequent labels in figure **??**

3.2 Train, Validation, and Test Sets

Our approach randomly selects 10% of the complete dataset for validation, and another 10% for testing. Since the data points were selected at random, we believed that the three sets are generally from the same underlying distribution. Hence, we settled for picking at random from the underlying training data set, which is unlike the approach described in the related work above.

3.3 Data Preprocessing

The two main data preprocessing areas were

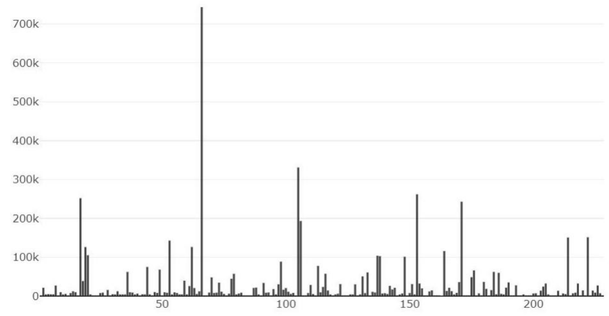
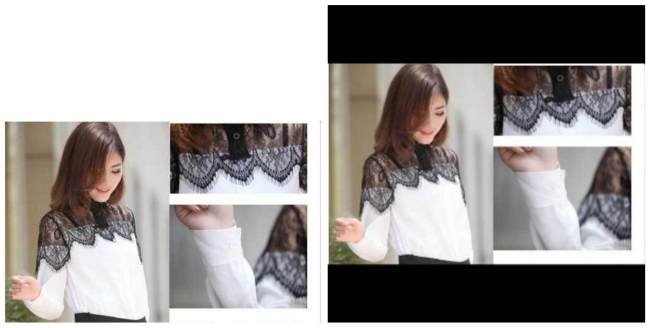


Figure 2: Histogram of label occurrences in the complete dataset

1. Resizing the original images
2. Padding images so they are of uniform shape

This way the images have uniform shape and fit the pretrained model. Figure 3 illustrates this.



(a) Original image (b) Preprocessed image

Figure 3: Comparison between an original and preprocessed image

4 Methods

We applied transfer learning using the Inception v3 network pre-trained on ImageNet. Inception v3 is trained for single-label classification among 1,000 classes, using a softmax output layer. To work with multi-label classification, we propose to replace the softmax layer with a fully connected layer with 228 outputs followed by sigmoid activations to detect each label. We combined the approaches of Bartyzal (2017) and Dziedzic & Miziuła (2017).

To retrain the last layer, we experimented with both binary cross entropy loss,

$$L = \sum_{i=1}^{C} [-y_i \log q_i - (1 - y_i) \log(1 - q_i)] \tag{1}$$

and weighted binary cross entropy loss,

$$L = \sum_{i=1}^{C} [-\text{weight} \cdot y_i \log q_i - (1 - y_i) \log(1 - q_i)] \tag{2}$$

where

$$q_i = \frac{1}{1 + \exp(-z_i)}$$

We have leveraged our Google Cloud credits to use 2 GPUs to train our model, and get more storage capacity on our instance to manage the data.

5 Results

5.1 Performance

We used mean F1 score, so both precision and recall are equally important. Remember that F1 score is given by

$$F1 = \frac{2 * precision * recall}{precision + recall} \tag{3}$$

Table 1 below summarizes the results of our various tests.

5.2 Hyperparameter Tuning

We trained with several learning rates; 0.05, 0.1, 0.5 and 1. We also tried different numbers of steps, with different sizes of datasets. One of our first findings is that learning performed slightly, but not significantly, better with 0.1 learning rate. From there, we tried different combinations of data set size, training steps and weights in the cross entropy loss function. Weight of 4 or 6 has shown to lead to higher F1 scores. The F1 score was still improving at 22,000 steps and 1M images, so there is probably room for improvement by training the model longer. A summary of results is presented in Table 1.

Images	Training Steps	Weight	Val acc	Test Precision	Test Recall	Test F1
50	500	1	.941	1	.167	.286
50	20k	1	.929	.500	.167	.250
100k	20k	1	.979	.846	.198	.320
100k	20k	6	.966	.370	.526	.434
100k	5k	4	.972	.458	.372	.411
1M	500	1	.976	.746	.125	.215
1M	500	4	.969	.374	.339	.355
1M	5k	4	.972	.439	.349	.389
1M	22k	4	.973	.474	.434	.453

Table 1: Training results

5.3 Network Visualization

First, we visualized the learned weights of our model. For example, in figure 4, we see the weights of the first convolutional layer. This shows the colors and edges that each filter will detect. For example, the filter that is mostly red, will focus on image content in the red channel and discard content in the green and blue channels.

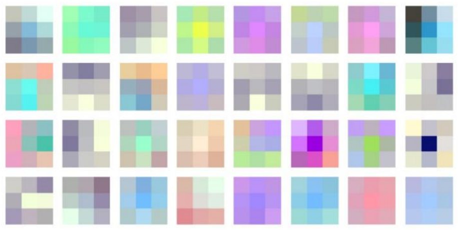
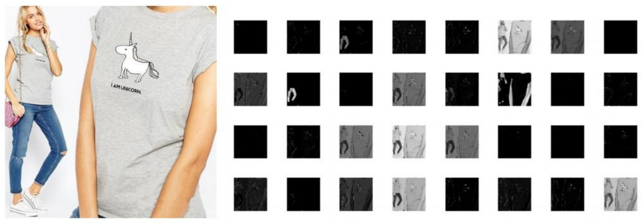


Figure 4: Visualization of our trained network's first convolutional layer

Second, we looked at the output of the activations of each filter in the first convolutional layer as well. In figure 5, we can see the activation output of each filter leads the network to focus on a specific part of the input image. For example, we can see that some filter activations focused on the jeans in the original image.



(a) Original image (b) Activations of the first convolutional layer

Figure 5: Each filter focuses on specific features of the original image

6 Conclusion & Future Work

By applying transfer learning techniques, we were able to train a network and produce meaningful predictions. Transfer learning notoriously saves computation costs that can pile up, especially during the tuning process.

An appropriate choice and adaptation of the loss function was shown to be key in improving performance. Different precision-recall tradeoffs were experimented until a good configuration was found.

Future work could further improve this model's performance. With more time, team members and computational resources, our recommendations for exploration would be training different final layer architectures, experimenting different hyperparameters training the full data set, and generating more visualizations to understand the network's learning.

7 Contributions

The authors contributed equally to the project. The authors acknowledge Klemen Cas, co-author of the shared project for CS230: Deep Learning course, Winter 2018, at Stanford University.

References

Radek Bartyzal. Multi label inception network. `https://https://github.com/BartyzalRadek/Multi-label-Inception-net`, 2017.

Krzysztof Dziedzic and Patryk Miziuła. How to create a product recognition solution. `https://blog.deepsense.ai/how-to-create-a-product-recognition-solution/`, 2017.

Y. LeCun, Fu Jie Huang, and L. Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 2, pp. II–97–104 Vol.2, 2004.

Qi Qian, Rong Jin, Shenghuo Zhu, and Yuanqing Lin. An integrated framework for high dimensional distance metric learning and its application to fine-grained visual categorization. *CoRR*, abs/1402.0453, 2014. URL `http://arxiv.org/abs/1402.0453`.

Sban. imaterialist(fashion): Eda+object detection+colors. https://www.kaggle.com/shivamb/imaterialist-fashion-eda-object-detection-colors, 2018.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567, 2015. URL `http://arxiv.org/abs/1512.00567`.

Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. *CoRR*, abs/1311.2901, 2013. URL `http://arxiv.org/abs/1311.2901`.