Apparel Classifier and Recommender using Deep Learning

Live Demo at: http://saurabhg.me/projects/tag-that-apparel

Saurabh Gupta
UC San Diego
sag043@ucsd.edu

Siddhartha Agarwal
UC San Diego
siagarwa@ucsd.edu

Apoorve Dave
UC San Diego
a1dave@ucsd.edu

ABSTRACT

We present a scalable approach for automatically detecting and tagging clothing products along with their colors, given a single image without meta-data. This has several interesting applications, including e-commerce, on-line advertising etc. Our pipeline makes use of the state-of-the-art methodology of extracting deep features using Transfer learning. The core of our process is a multi-class learner based on Logistic regression. To automate the whole process, we developed a web application which detects the clothing type and color of any query image. We have achieved a good clothing detection performance (86%), while being quite fast. Finally, we present a clothing suggestion scenario, where similar items from our dataset are presented to the user.

We have also provided a running service for the above task as a prototype of our work. The URL has been mentioned in the header.

Keywords

Clothing Suggestion, Clothing Detection, Color Detection, Automatic Image based Product Recommender

1. INTRODUCTION

Today's product recommendation for clothing items depends a lot on the similarity (Cosine/Jaccard) of the products, which in turn depends on the users who viewed/bought those products, rather than the visual aspects/similarity of the products.

Our aim for this project is to identify from an image, its apparel type and color, as well as recommend similar products from our dataset. To achieve this, we transformed the raw images using a pre-trained neural network, to get the deep features of images which are then used to train Logistic Classifiers. We have achieved good accuracy in our predictions of the cloth type and color.

Our work is organized as follows: Section 2 describes the dataset used. Section 3 discusses the present day methods and our own method to approach this problem. Section 4 gives a detailed discussion on the model we used, along with what worked and what did not work. Section 5 is a summary of our results and conclusions regarding our approach, with suggestions on future works and improvements.

2. DATA SET USED

The training-data was collected by downloading clothing images from the web. Google image search API was used to collect 100 images for each Clothing Style x Color label combination making it a total of 10000 labeled images. We could not use a bigger dataset because of the upper limit imposed on number of hits per day by the API. We used a random 60-20-20 split as train-validation-test datasets.
Clothing types and color labels used to perform the queries are as follows:

Table 1: Cloth/Color Labels

Cloth Style		Color		
shorts	shits-tshirt	black	white	blue
suit-blazer	skirt	brown	yellow	pink
jeans-trousers	hats	gray	green	purple
		orange	teal	red

As the query results consisted of images from many different sources, our dataset contained images with different types of backgrounds, angles, orientations etc. For example, a search for blue trousers returned some images with just the trousers with white background, some with models in them, and some with real life backgrounds like parks etc. Some of the outliers were also present in the dataset which matched, say, just 'trousers' and returned, a different colored trouser, creating some noise in our training set.

3. LITERATURE REVIEW

Object recognition is a well known challenge in machine learning and computer vision. Many algorithms exist for performing these tasks, but selecting a scalable and efficient model has always been tough.

3.1 Image Processing based Classifier

Intuitively, many algorithms use methods like 'edge detection', 'variation in image locally', 'region segmentation' etc. to extract features/properties for training the predictive model.

Though these features may work good for a certain class of objects, often such features are not present in different classes. For example, if we use a 'line-segment detection' based model to predict sports equipment, it may work efficiently for objects like a bat, wicket or a football-post. But if we introduced a new object class for balls, the model's

efficiency will be drastically affected.

Such methods are also prone to distortion, luminosity, angle etc. of the object's image.

3.2 Deep Learning based Object Recognition

Deep Learning uses several layered neural network with millions of parameters, to fit the observed data. It uses a composition of multiple non-linear transformations in contrast to standard ML methods like linear classifier or decision tree based classifier, and hence provides a better accuracy.

Deep Learning is believed to be the best known model for many problems including Object Recognition. Deep neural networks are able to detect, classify and describe objects appearing in natural scenes, better than the hand crafted feature extractors like HoG, SIFT, SURF.

used as well.

3.3 Transfer Learning

Transfer learning is the improvement of learning in a new task through the transfer of knowledge from a related task that has already been learned.

Instead of training a deep neural network similar to the AlexNet, we switched to Transfer Learning Model. We used the initial layers of the neural network used by AlexNet directly to extract the Deep Features for the clothing images. As such, our model changed from training the parameters of the neural network to training the parameters of simple classifiers making use of the extracted deep features. This saved us a lot of computational power and drastically reduced the size of training dataset. Our model complexity was quite simplified as deep features have proved to be ex-

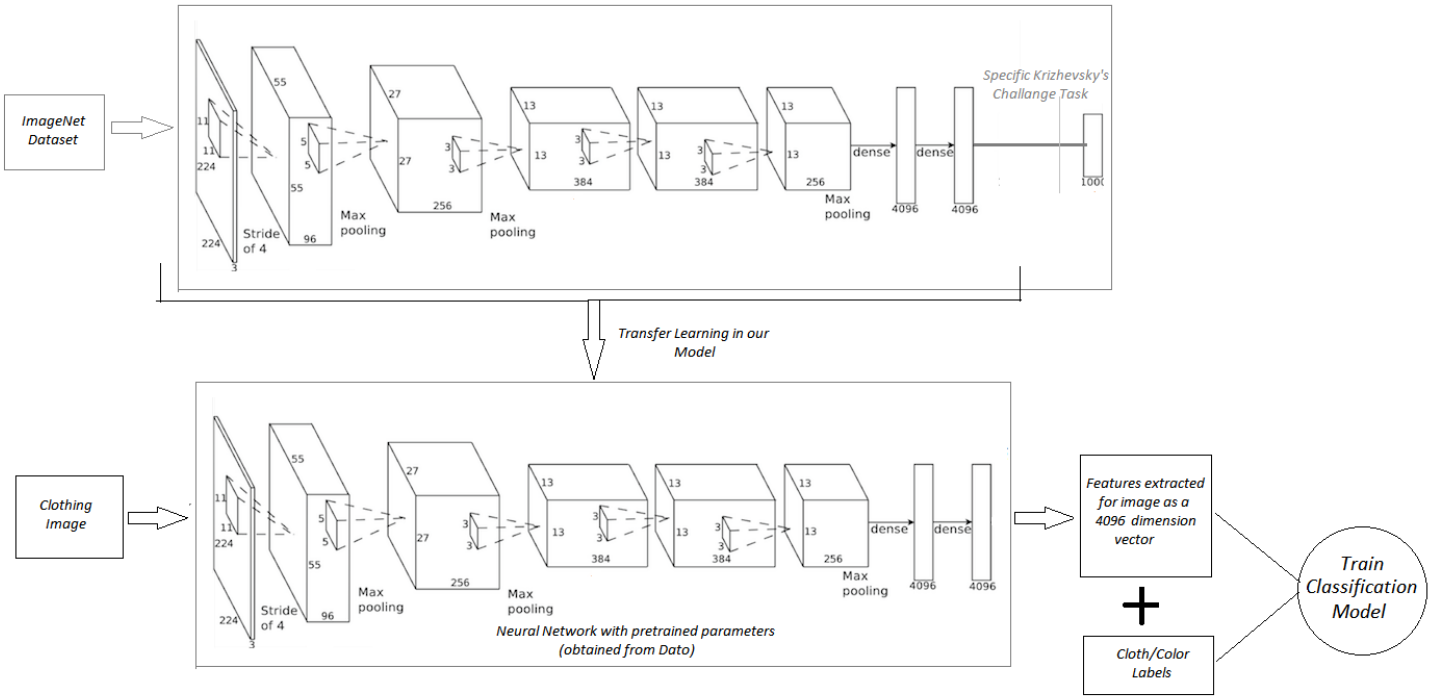

Figure 1: Model Used

AlexNet Deep Network Convolution model

AlexNet [2] is a now a standard architecture known in the deep learning community and often used for bench-marking. This Deep Neural Network model used in ImageNet's LSVRC-2012 contest [2] was trained on 1.2 million images spanning over 22000 categories, and is now commonly known as AlexNet. The DNN had over 60 million parameters and 650,000 neurons. It was 7 hidden layers deep, where the first five layers were convolution layers.
To avoid over-fitting, the DNN was trained on 224px x 224px random patches, and horizontal reflection of each image was

cellent predictors for identifying patterns in images.

The first five layers in this model were more generalized over the images, but the last three layers of the network were task specific to the contest's requirement. We used the output of layer seven which is a 4096 dimensional vector representation of the Deep Features extracted from the original(256 x 256) dimensional image, to train the classifier.

This model is used as described in fig.1 to extract features from images and train the classifier.

4. OUR MODEL

We focus on identifying the type and color of clothing that is displayed in images on e-commerce websites. This demands the use of a pre-trained deep neural network to extract the deep features from the query image which are then fed to two different classifiers for identifying clothing type and color.

4.1 Preprocessing and deep features extraction

The query image is scaled to 256 X 256 pixels since the original Deep Neural Network was trained on images of this size. The neural network transforms the query image to give a set of 4096 deep features which are then used for classification.

4.2 Clothing Type Classification

We started by defining the clothing styles that our model would detect. The training dataset consisted of **6,000** labeled images equally distributed over the defined classes. To create the classifier, we looked at 3 models: Logistic Regression, Random Forests and Boosted Trees.

Model Selection and Parameter Tuning

Out of the above 3 models(with base settings), Logistic Regression outperformed the decision tree based methods. The performance of all the models is described below:

Table 2: Model Comparison

Model	Training Accuracy	Validation Accuracy
Boosted Trees	.98	.81
Random Forests	.92	.80
Logistic Classifier	.90	.83

On fine-tuning the above models and checking for overfitting, Logistic classifier gave even better performance of 86% accuracy. It turns out that the deep features are very sparse and linear models work better than the desicion trees based methods in this case.

Logistic classifier Parameters

To tune the parameters for Logistic classifier, we used 5-fold cross validation methodology with grid-search over the following parameters:

params = {
'*max_iterations*' : [5,10,15,25],
'*regularization_parameter*' : [0.1, 0.05, 0.01]

We quickly ran into scaling issues to perform cross validation. With increasing number of iterations, the computations required more memory and computing power. To cope with this, we moved to a more powerful AWS compute server.

Results of above grid search are summarized in the figure below:

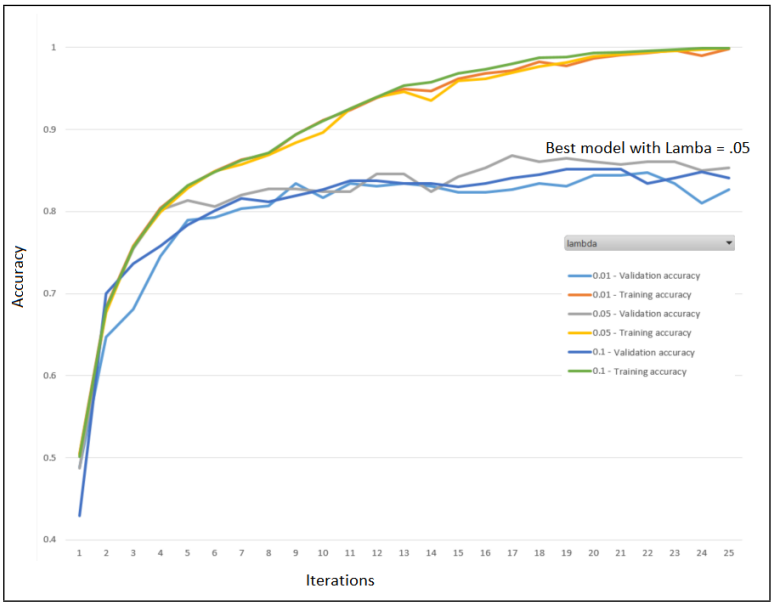

Figure 2: Tuning Logistic Regression

Hence, we finally used the model with regularization parameter of .05 and performed 25 iterations to get 86% accuracy.

4.3 Color Detection

For color classifier, we used the exact same approach as the clothing type classifier. Of the three models, the Logistic classifier performed the best with 71% accuracy on 5-fold cross validation dataset with a regularization parameter of .008. The color classifier had 69649 total coefficients as opposed to 20485 in the clothing type classifier, and thus each iteration took even longer.

4.4 Visual Recommendations

To generate recommendations based on visual similarity of query image, we created a k-nearest neighbors model which recommends top k similar images based on euclidean distance calculated using deep features of images. We could not come up with a good approach to evaluate this recommender system other than manually eyeballing the results.

4.5 Unsuccessful attempts

1. Instead of training our classifiers on deep features(using transfer learning concept), we started with creating a deep neural network from scratch. We concluded that it is extremely difficult to fine-tune a deep neural network given the resource limitations - time, data and computational power. We also tried to train neural network using AWS G2 instances (having high-performance NVIDIA GPUs, with 1,536 CUDA cores), but our dataset was too small to create an accurate neural network with over a million parameters, so it was a good decision to use deep features from a pre-trained neural network by AlexNet [2](which was trained using 1.2 million images).

2. Instead of creating two different classifiers for colors and clothing types, we tried creating a single classifier to describe the clothing (for eg. "blue shirt"). However, even after tuning the model, the classifier could predict the joint label with just 62% accuracy. The reason for such a low accuracy can be attributed to the fact that the original neural network generates deep features to differentiate between objects of different shapes rather than different colors. Hence, the deep features being used were not very helpful in predicting the right colors.

5. RESULTS AND CONCLUSIONS

To test the model performance, 20% of the dataset was held out as test set. Table 3 summarizes the performance of Logistic classifier which was used to predict clothing styles. **Table 5** is the confusion matrix showing right and wrong predictions for each clothing type class.

Final model scores for clothing type classifier:

Table 3: Type Classification Scores

Accuracy:	0.8605388272583201
F1 score:	0.862446649188848
Recall:	0.8628132488915848
Precision:	0.8625174047786341
AUC:	0.9844942204579811

Final model scores for clothing color classifier:

Table 4: Color Classification Scores

Accuracy:	0.7123613312202852
F1 score:	0.6161657571038678
Recall:	0.6143156337769639
Precision:	0.6349581719304283
AUC:	0.9623511969918295

5.1 Discussion

It turns out that the clothing type classifier performs really well giving a performance similar to the original AlexNet model(85% accuracy) that generated features to differentiate between 22000 different classes.
For clothing classifier, one of the main issues was in the training dataset where the same image consisted of clothing of different colors, for eg. white shirt and brown pants. Giving a single color label to such images, for eg. brown in the above example, might have introduced noise in the training process leading to poor performance of the color classifier.

5.2 Future Works and Improvements

We realized that the above method of data collection - Using search terms like "blue shorts" on Google images and then downloading the images was not the best thing to do. The search, in many cases, returned images that were not in agreement with the search term and such mislabeled images in the training dataset might have caused the model to perform less accurately. This methodology can be improved by either collecting the labeled clothing images from some other reliable data source or manually checking the labeled images - which comes at the cost of time and money.

For classifying the color, less complicated image processing techniques might be able to do this job better than the method described in this report. The deep features that were generated to differentiate between objects of different size and shape might not be very effective in differentiating between different colors. Also, images containing clothing of different colors should be labeled appropriately - they can be split into multiple images by segmentation or edge detection and can then be fed to the clothing type classifier.

Table 5: Cloth Type prediction on test

target_label	predicted_label	count	is true
tshirt-shirt	tshirt-shirt	342	1
hats	hats	203	1
skirt	skirt	209	1
suit-blazer	suit-blazer	320	1
jeans-trousers	jeans-trousers	447	1
shorts	shorts	198	1
suit-blazer	tshirt-shirt	19	0
jeans-trousers	suit-blazer	38	0
hats	tshirt-shirt	3	0
suit-blazer	jeans-trousers	25	0
tshirt-shirt	hats	3	0
jeans-trousers	tshirt-shirt	8	0
skirt	shorts	22	0
hats	shorts	3	0
hats	suit-blazer	2	0
jeans-trousers	skirt	8	0
skirt	suit-blazer	2	0
skirt	jeans-trousers	16	0
tshirt-shirt	jeans-trousers	6	0
shorts	suit-blazer	6	0
shorts	skirt	22	0
hats	jeans-trousers	2	0
shorts	jeans-trousers	16	0
suit-blazer	shorts	13	0
skirt	tshirt-shirt	2	0
jeans-trousers	shorts	25	0
hats	skirt	2	0
tshirt-shirt	shorts	2	0
tshirt-shirt	suit-blazer	21	0
shorts	tshirt-shirt	5	0
suit-blazer	skirt	5	0
tshirt-shirt	skirt	5	0
		2000	1719
	Accuracy =	$\frac{1719}{2000}$ =	0.859

Currently, the model can recognize only one type of clothing and color in the query image. Various image segmentation techniques can be used to split images into candidate products and can then be fed to the classifiers to recognize all clothing types and colors in a given image.

To generate more value out of this project, more data is needed to differentiate between more clothing types and other classes such as persons(child, boy, girl, male, female), pattern of clothing(floral, dotted, checkered, striped), materials(cotton, denim, leather) and styles(summer, autumn, winter, business, casual). All of these classes can be very helpful in generating metadata for clothing products and in generating fashion recommendations which can have a substantial impact on the market and fashion industry.

6. REFERENCES

[1] Apparel classification with style. http://people.ee.ethz.ch/~lbossard/projects/accv12/index.html. (Visited on 12/02/2015).

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *NIPS*, 1(1):795–825, November 2012.

[3] M. F. e. a. Stephan M. Parameterized shape models for clothing. http://www.cs.berkeley.edu/~pabbeel/papers/MillerFritzDarrellAbbeel_ICRA2011.pdf. (Visited on 12/02/2015).

[4] L.-J. L. Yannis, Lyndon. Getting the look: Clothing recognition and segmentation for automatic product suggestions in everyday photos. *JCMR'13*, 2013.