CS230

Search My Favorites by Color: Fashion Parsing through Color Classification

Calvin H. Guo
Department of Computer Science
Stanford University
calving@stanford.edu

Abstract

This paper investigates using computational color naming approach that mapping pixel values from fashion images to standardized color names. Current commercialized solutions are all based on vendor-generated linguistic tags such as "red sweater", "black jeans" putting into the HTML meta data, and letting user perform text searches based on those tags. In this work, we consider using a convolutional neural network (CNN) and softmax activation to classify the color of fashion images. Since the color naming only focused on ROI (region of interest) in the image, we use Region-based convolutional neural network (R-CNN) to focus on pixels within the detected boxes as the training input, and use softmax for classification and learn the fashion colors. More specifically, we adopt the Mask R-CNN with the Semantic Segmentation approach to adapt to the fashion images. This approach can let users filter their favorite fashions by a standardized color naming without the help of linguistic color tags from the online images.

1 Introduction

We propose a deep learning approach to learn the color of fashion images. Understanding the color of fashion images can allow customers filtering their favorite cloth based on a selected color. The usage scenario could be in a huge amount of fashion images data store, the customer would like filter those online images (collected from multiple vendors/websites) by a specific color category. For example, when a customer searches a sweater, and then want to filter by 'Red' sweater, our backend deep learning service will have all the pre-analyzed sweater images within the Red category returning back, showing as an example in Figure 1.

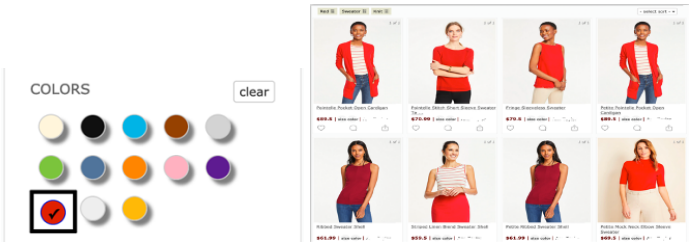

Figure 1: Customer filter sweaters with 'Red' color to showing up

Even though most of the online stores have the color filtering functions, but those are implemented by linguistically labelling approach through HTML tag, instead of auto-classification through a

CS230: Deep Learning, Winter 2020, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

computational technology. Also, the color tag could be vague, such as 'vanilla', 'peach', 'olive', 'navy', etc., the name convention for different vendor have different and randomly defined color names. Sometimes using HTML tag could be inaccurate if the product have multiple colors available. For example, the tag includes 'red', 'white' and 'blue' in html metadata, but the actual product image only showing up 'blue' on the webpage. Thus, we need to define a standardized color category, and based on that analyze fashion image itself to get a standardized color category. We also facing the challenge that the shape of cloth could be variable and need to wipe out the background color.

To resolve those issues, we utilized the latest version of R-CNN [1], Mask R-CNN [2], to identify cloth colors. Mask R-CNN is derived from Faster R-CNN [3] approach with semantic segmentation capability, which is a task of classifying each and very pixel in an image into a class. The granularity of understanding at pixel level is very suitable for our goal since we are mostly focused on colors of pixels in the region of the cloth. We can put more weights to the pixels in the object boundary than the surrounding pixels when learning the color. Current classification researches through CNN are mostly focused on types of objects (person, vehicles, animals etc.), or if used in fashion domain, they are on identifying the types of cloth (shirts, jacket, pants etc.). In this work, this is the first experiment to use CNN, or more specifically - a Mask R-CNN approach, to make classification of colors of objects instead of the types of objects.

2 Dataset and Features

Fashion-MNIST [4] is one of the most popular dataset for fashion deep learning, but it does not apply to us because it's mainly used for training the category of cloth and all are grey-scale images. The DeepFashion [5] is more suitable for us on such Color-Labeling project. It has over 800,000 diverse fashion images and rich annotations with additional information about landmarks, categories, pairs etc. The dataset consists of 5 different kinds of predicting subsets that are tailored towards their specific tasks. In our project, since we are more focused on shopping website images from different vendor/department store, we will use images in the "In-shop Clothes Retrieval Benchmark" as our training dataset. That dataset has the most similar patterns and styles with our target shopping images.

However, the format of the annotations of DeepFashion is not compliance of our project. In this project, we leverages Detectron2 [6] project in our implementation, which able to recognize objects represented by the classes from the COCO (Common Object in Context) dataset [7]. Thus, we will transform from JSON annotation format of DeepFashion into COCO format as data inputs, which will be more detailed explained in later sections.

3 Preprocessing

An issue of existing color naming in the dataset is using vague or subjective names while not follow a standard name convention. To align with our purpose of color prediction, first, we want to standardize the color annotation of those images into one of those 10 broad categories: *1: Black; 2: Blue; 3: Green; 4: Magenta; 5: Pink; 6: Purple; 7: Red; 8: Yellow; 9: White; 10: Navy (Deep Blue)*. Next, encode the colors as a vector to represent each type. For example, if the output $Y = [[0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0]]$ which means the output color is Green. To give a smoothly unified training process with fixed input size X in the training dataset, we will resize all image by 256×256. The image will use RGB encoding, thus the input vector will be a fixed vector size of $256 * 256 * 3 = 196608$ elements for each training example.

After standardize the color name convention of all the training examples, we need to generate the JSON file with COCO format [7] as the date input. Here we used the LabelMe tool [8] to generate boundary annotation of the training examples, and then using a python script (labelme2coco.py) [9] to transform the JSON annotation generated by LabelMe to COCO format. After that, we can directly input out training dataset to Detectron2 API [10].

4 Methods

Since we only interested in the color of the cloth which is a region of interest (ROI) in the picture, we start the idea of Region-based convolutional neural network (R-CNN) architecture to train the data. The goal of R-CNN is to take in an input image, correctly identify where the main objects (via

a bounding box detected by Object Detection algorithms) in the image, and then use CNN to train the region in the bounding box only. The general R-CNN are used to detecting the types of objects. Since we are interested in classifying the color of objects, more specifically, the color naming process for fashion image needs to (in Figure 2):

1. Generate a region proposal using Object Detection or Selective Search algorithms used as a bounding box to generate Region of Interest (ROI).
2. Standardize the size of proposed region in the image and pass it through a pre-trained CNN or modified version to be used as the feature extractor
3. Using softmax regression model to output the classified category of the interested object. Here we are focus on the normalized color of the interested object.

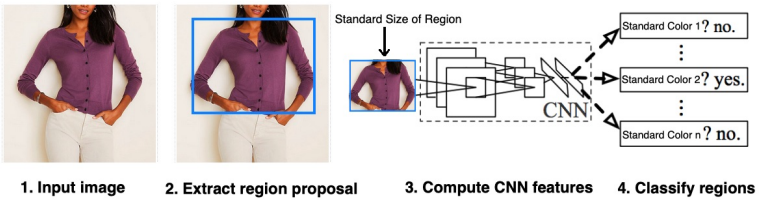

Figure 2: R-CNN: training detected region with CNN features

Based on the idea of R-CNN, there are two advanced version, Faster R-CNN [3] and Mask R-CNN [2]. Instead of generating a bunch of potential regions using Selective Search [11], Faster R-CNN uses a single CNN to both carry out region proposals and classification. First, the image is provided as an input to a convolutional network which provides a convolutional feature map. Then, a Region Proposal Network works by passing a sliding window over the CNN feature map and at each window, outputting k potential bounding boxes and scores for how good each of those boxes is expected to be. The predicted region proposals are then reshaped using a RoI pooling layer which is then used to classify the image within the proposed region. We utilized the Detectron2 [6] framework with model-zoo package. In the package, it adopts classical AlexNet [12] as CNN to achieve the goal of feature mapping and classification. It's architecture showing as Figure 3.

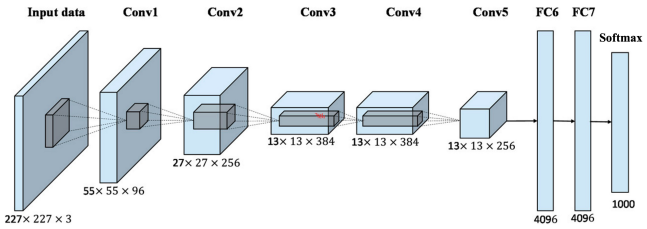

Figure 3: AlexNet: Consists of 5 convolutional layers and 3 fully connected layers

Since we classifying output as 10 categories of colors, the output layer will use Softmax activation with 10 classifications, i.e., $\sigma(Z)_i = \frac{e^{Z_i}}{\sum_{j=1}^{10} e^{z_j}}$, where Z is the linear output from softmax layer.

We also adopt the latest Mask R-CNN implementation in Detectron2 to accurately training and detecting the colors at pixel level. Mask R-CNN is an extended version of Faster R-CNN for pixel level segmentation. Mask R-CNN does this by adding a branch to Faster R-CNN that outputs a binary mask that says whether or not a given pixel is part of an object. The branch is just a Fully Convolutional Network (FCN) on top of a CNN based feature map. Here are its inputs and outputs:

- **Inputs:** CNN Feature Map.
- **Outputs:** Matrix with 1s on all locations where the pixel belongs to the object and 0s elsewhere (this is known as a binary mask).

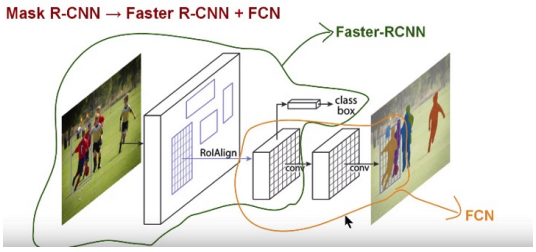

Figure 4: The Mask R-CNN framework

In summary Mask R-CNN combines the two networks — Faster R-CNN and FCN in one mega architecture. Figure 4 shows the conceptual diagram. It adds a branch at output of Faster-RCNN to output the object mask — a binary mask that indicates if the pixels were in the object bounding box. This branch is a small FCN applied to each RoI, predicting a segmentation mask in a pixel-to-pixel manner. Mask R-CNN is simple to implement when the Faster R-CNN framework was given, it very suitable to our goal of color classification for fashion images and thus we adopted such approach.

5 Experimental results and observations

Figure 5: Examples of test dataset results: pure colors

We generated around 200 images with COCO format boundary annotation as training dataset input, and iterates 450 rounds generating a good enough weights and hyper-parameters. Figure 5 shows the examples of color prediction results from test dataset with pure color cloth. We can see Mask R-CNN approach gives a very satisfied boundary detection of the cloth. The detected color is also satisfied. Except for black color, all colors return a confidence probability of above 75 percentage, and all gives more than 55% of confidence of correct color prediction. Figure 6 shows the color prediction results for non-pure color (e.g., striated, dotted) cloth. Surprisingly it shows good results as well. All gives corrected/make sense colors based on the background or highlighted colors for non-pure color clothes. Except for black color, all gives a high confidence of probability in prediction.

Definitely there are some issues need improvements and investigation. In Figure 7, (a) showing the case when the color between magenta and red, and then it gives 68% of magenta and 49% of red with multiple boundaries. (b) gives a correct prediction of magenta color cloth but also detected another boundary of face with pink color, which do not make sense as our purpose only focus on fashion color. (c) basically did a good job to predict the correct boundary in a multi-color cloth case as it detects the major color, but it hard to say it should be marked as navy (deep blue) or black. (d) is not a correct color prediction. We believe it could caused by inconsistent labeling of such color in training set, may paradoxically labeling as purple, blue or navy at different examples, causing the training model is confused. Table 1 shows color prediction statistics for each color. We note that the

black color prediction needs improvements in our model, while all others form a good shape. We think this is because more colors such as navy, magenta and deep purple are easy to confuse black color. We will consider making a more consistent and standardized labeling of deep colors in our training set to let the model more easily differentiate out the black with other colors.

Figure 6: Examples of test dataset results: non-pure colors

Figure 7: Issues of some observations

Color List		
color	confidence probability	accuracy
Black	51%	77%
Blue	91%	100%
Green	85%	94%
Magenta	71%	85%
Navy	73%	86%
Pink	94%	100%
Purple	89%	92%
Red	92%	100%
White	88%	99%
Yellow	88%	98%

Table 1: Prediction statistics for each color

6 Conclusion/Future Work

In this paper, we presented an approach of color prediction for fashion images using the Mask R-CNN technique, which can provides pixel level of instance segmentation of images. All previous classification works through convolutional neural network (CNN) and including Mask R-CNN are all focused on types of objects. This is the first experiment for classifying color of objects by CNN as far as we know. It makes satisfied and expected results. There are some issues need improvements for particular colors, we think giving a consistent labeling for various colors in training dataset is important to improve the prediction performance. We will be more cautious on color labelling and in future we could investigate the best hyper-parameters, weights and iteration loops for different sizes of training set to achieve the optimized performances.

References

1. Girshick, R., Donahue, J., Darrell, T. & Malik, J. *Rich feature hierarchies for accurate object detection and semantic segmentation* in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2014).
2. He, K., Gkioxari, G., Dollár, P. & Girshick, R. *Mask R-CNN* in *Proceedings of the International Conference on Computer Vision (ICCV)* (2017).
3. Ren, S., He, K., Girshick, R. B. & Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **39,** 1137–1149 (2015).
4. Xiao, H., Rasul, K. & Vollgraf, R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747* (2017).
5. Liu, Z., Luo, P., Qiu, S., Wang, X. & Tang, X. *DeepFashion: Powering Robust Clothes Recognition and Retrieval with Rich Annotations* in *CVPR* (IEEE Computer Society, 2016), 1096–1104. ISBN: 978-1-4673-8851-1.
6. Wu, Y., Kirillov, A., Massa, F., Lo, W.-Y. & Girshick, R. *Detectron2* `https://github.com/facebookresearch/detectron2` (2020).
7. Lin, T.-Y. *et al. Microsoft COCO: Common Objects in Context* in *ECCV* (2014).
8. Russell, B. C., Torralba, A., Murphy, K. P. & Freeman, W. T. LabelMe: A Database and Web-Based Tool for Image Annotation. *International Journal of Computer Vision* **77,** 157–173 (2007).
9. Zhang, C. *How to create custom COCO data set for instance segmentation* `https://www.dlology.com/blog/how-to-create-custom-coco-data-set-for-instance-segmentation/` (2020).
10. Lab, F. A. *Detectron2 API Documentation* `https://detectron2.readthedocs.io/modules/index.html` (2020).
11. Uijlings, J., van de Sande, K., Gevers, T. & Smeulders, A. Selective Search for Object Recognition. *International Journal of Computer Vision.* `http://www.huppelen.nl/publications/selectiveSearchDraft.pdf` (2013).
12. Krizhevsky, A., Sutskever, I. & Hinton, G. E. *Imagenet classification with deep convolutional neural networks* in *Advances in neural information processing systems* (2012), 1097–1105.