Deep Shopping: Fashion Recommendation Based on Object Detection

Alexia Wenxin Xu
alexiaxu@stanford.edu

Fei Liu
fliu5@stanford.edu

Xiaoyan Wu
xw1@stanford.edu

Abstract

With the rapid expansion of E-commerce, people have developed higher shopping enthusiasm and increasing demand for garments. In many online shopping websites, customers can conveniently search their favorite clothing by keywords. However, there is no available fashion image retrieval system based on the customer's query image. In this project, we aim to recommend fashion items similar to the ones in the query image. We re-formulate the task as an object detection problem. By training a Faster R-CNN to classify and localize the clothing, we force the model to not only recognize the fashion item in the image but also capture its category and other features. Our model achieves an accuracy of 99% *in classifying the clothing items and an* $MAP@0.5$ *of 0.725 in detecting bounding boxes. Our model was able to detect clothing both in the images from the dataset and in online random images. Amazingly, our model is able to recognize the design, fabric, pattern, and color of the query clothes to provide precise recommendations. We hope our project could be a first attempt at the commercialization of fashion retrieval systems.*

1. Introduction

Fashion industry occupies a significant position in the global economy. With the rapid expansion of E-commerce, algorithms facilitate online fashion shopping have drawn more and more attention. Promising progress has been made on clothes classification [1, 22], attribute prediction [12, 18] and fashion item retrieval[20, 13]. Customers now enjoy high-quality recommendations by searching websites with keywords. However, it is still inconvenient to find similar clothing items to street snaps or Instagram photos. Shopping systems with such functions would create huge business value.

Previous efforts have been made to fill in the blank. Liu et al. [14] propose a cross-scenario clothing retrieval system by deriving the similarity of images using direct sparse reconstruction. Kang et al. [11] implement a fashion recommendation and design system by training the image representation and the recommender system jointly. Jaradat et al. [10] explore domain adaptation for efficient recommendations. Nevertheless, current models are not satisfying enough in capturing the deep features of the images, such as the fabric, pattern, and color of the clothing items.

Object detection is one of the key tasks in computer vision. It involves detecting instances of objects from a particular class in an image. In 2015, Girshick et al. [5] propose Fast R-CNN, a clean and fast update to R-CNN and SPPnet with the state-of-the-art detection results. A year later, Ren et al. [17] unify Region Proposal Networks (RPNs) with Fast R-CNN [5] and enable nearly cost-free region proposals. We base our fashion retrieval system on Faster R-CNN and try to achieve better deep feature retrieval.

In this project, we aim to recommend clothing to customers based on the uploaded query image. We trained Faster R-CNN to localize and classify the clothing items despite the variance in postures of the persons in the images. The input to our algorithm is an image. We then train Faster R-CNN [17] to output a predicted class and bounding box. We extract the feature maps before the RPN as the deep features of the images and conduct k-nearest neighbor search based on those features. Our model was able to detect the class and location of fashion items in real life images. It could also extract the fabrics, patterns, and colors of the query image and provide high-quality recommendations.

2. Related Work

Fashion Dataset: Many inspiring previous works contribute to a better fashion shopping experience. Liu et al. [15] collect the DeepFashion dataset that is used in our project. It is a large-scale clothes database with over 800,000 diverse fashion images ranging from well-posed shop images to unconstrained consumer photos. Kiapour el al. [6] and Huang et al.[9] also collect datasets in smaller scales and label images with attributes, bounding boxes, or consumer-to-shop pair correspondences. These works provide researchers abundant image data for further exploration, and we choose the deepFashion dataset to train our own model.

Clothing Recognition: Before the prosperity of deep learning, clothing recognition mostly relied on handcraft features [21, 3]. SIFT [16] and HOG [4] are good ex-

amples of those models. It's not surprising that handcraft features are less expressive than the deep features we use nowadays. In recent years, a number of deep models have been introduced to learn more discriminative representation in order to handle cross-scenario variations, including Dual Attribute-aware Ranking Network [9] and CNN model [6]. In 2016, Liu et al. [15] also proposed a CNN-based deep learning model for the landmark detection of fashion images.

Object Detection: Object detection is the other relevant field with a large literature. Comprehensive surveys and comparisons of object proposal methods can be found in [8, 7, 2]. Fast R-CNN [5] and Faster R-CNN [17] are two impactful models that achieve the state-of-the-art performance. Fast R-CNN efficiently classifies object proposals using deep convolutional networks. Faster R-CNN introduces RPN that shares full-image convolutional features with the detection network, thus enabling nearly cost-free region proposals.

Observing that previous clothing retrieval models are relatively ineffective in extracting the deep features of clothes images, we re-formulate the task as object detection and employ Faster R-CNN to generate deep features. Our project is the first work to incorporate object detection into fashion recommendation.

3. Methods

In this section, we will explain our methods for fashion recommendation. We first start with a baseline model, then anatomize the structure of Faster R-CNN. At last, we will explain the feature extracted for recommendation.

3.1. Baseline

As a baseline, we first deploy a classification + localization approach and treat localization as a regression problem. Our baseline model is a five-layer CNN followed by a fully-connect layer. Similar to the model in lecture 11, there are two output heads connected to the fully-connected layer. The classification head outputs a softmax classification score, and the localization head outputs four box coordinates (x_1, y_1, x_2, y_2), which prepresents the top left and the bottom right corner of the object. We define the loss function as follows:

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n} softmax(s(\theta, \boldsymbol{X_i})) + \|\boldsymbol{b_i} - \hat{\boldsymbol{b}_i}\|_2^2 + \alpha\|\theta\|_2^2$$

where θ represents all trainable parameters. n is the number of examples. $\boldsymbol{X}$ is the input matrix. $\boldsymbol{b}$ is the ground truth of bounding box. $\hat{\boldsymbol{b}}$ is the predicted bounding box, and α is the regularization coefficient.

3.2. Faster R-CNN

Faster R-CNN (shown in Figure 1) is composed of two modules. The first module is a deep fully convolutional network that proposes regions, and the second module is the Fast R-CNN detector [5] that uses the proposed regions. The two modules are built into a unified system so that all parameters can be updated end-to-end through back propagation. Due to the complex structure of Faster R-CNN, we are only able to cover the main ideas in building the model. For more details, please refer to [17] or to our submitted code.

3.2.1 Region Proposal Networks

RPN (as shown in Figure 2) takes an image (of any size) as input and outputs a set of rectangular object proposals, each with an objectness score. To generate region proposals, a small window slides over the convolutional feature map output by the last shared convolutional layer. This small network takes as input an 3×3 spatial window of the input convolutional feature map. Each sliding window is mapped to a lower-dimensional feature. Slightly different from the original paper, we use 256-d for both ResNet-50 and ResNext-101. This feature is fed into two sibling fully-connected layersa box-regression layer (reg) and a box-classification layer (cls).

At each sliding-window location, our model predicts at most 9 region proposals. So the reg layer has 36 outputs encoding the coordinates of 9 boxes, and the cls layer outputs 18 scores that estimate probability of object or not object for each proposal. The 9 proposals are parameterized relative to 9 reference boxes, i.e. anchors. An anchor should be centered at the sliding window, and is associated with a scale and aspect ratio. 9 is the default value in [17], and it could be changed to other values as well.

Both anchors and the functions that compute proposals relative to the anchors share a great property: translational invariance. If one translates an object in an image, the proposal should translate and the same function should be able to predict the proposal in either location. The property also brings another advantage to the model. It reduces the model size by more than 100 times comparing to GoogleNet [19].

To train the RPNs, a binary class label (of being an object or not) is assigned to each anchor. Either one of the two types of anchors have a label of 1: (i) the anchor/anchors with the highest Intersection-overUnion (IoU) overlap with a ground-truth box, or (ii) an anchor that has an IoU overlap higher than 0.7 with any ground-truth box. Based on the idea, the objective function of the model is thus defined as:

$$L(\{p_i\}, \{t_i\}) = \frac{1}{N_{cls}}\sum_i L_{cls}(p_i, p_i^\star) + \lambda\frac{1}{N_{reg}}\sum_i p_i^\star L_{reg}(t_i, t_i^\star),$$

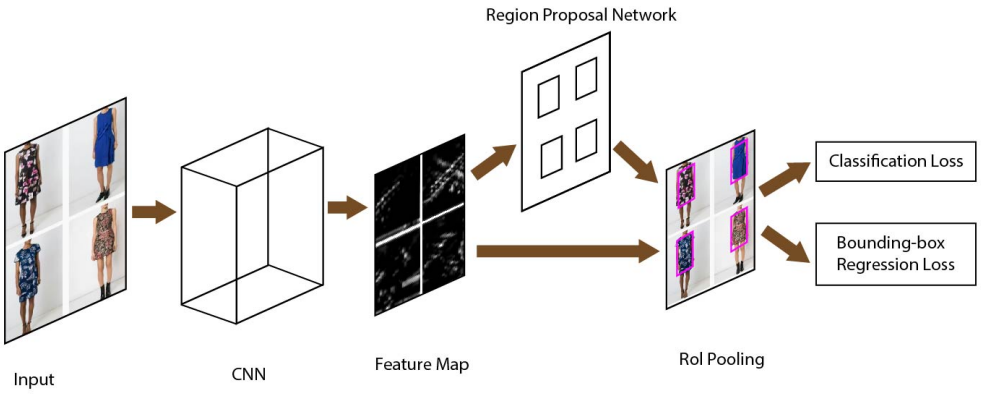

Figure 1. The Faster R-CNN model

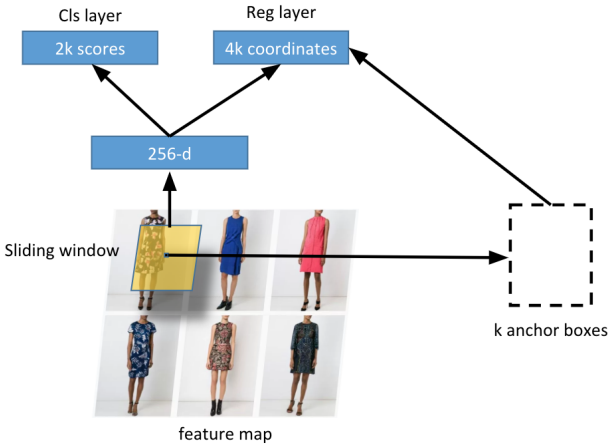

Figure 2. Structure of the Region Proposal Network (RPN)

where i is the index of an anchor in a mini-batch and p_i is the predicted probability of anchor i being an object. $p_i^\star \in \{0, 1\}$ is the ground truth. t_i is a vector representing the 4 parameterized coordinates of the predicted bounding box and $t_i^\star$ is the ground truth bounding box associated with a positive anchor.

To define the regression loss,

$$L_{reg}(t_i, t_i^\star) = R(t_i, t_i^\star),$$

where R is the smooth L_1 loss.

$$R_{L_1} = \begin{cases} 0.5x^2 & \text{if} |x| < 1 \\ |x| - 0.5 & \text{otherwise} \end{cases}$$

The smooth L_1 loss is less sensitive to outliers than L_2 loss and can prevent exploding gradients. The λ is a coefficient that balances the two loss and we use $\lambda = 1$ in our experiments.

The parameterizations of the 4 coordinates t_i are defined as follows:

$$\begin{aligned} t_x &= \frac{(x - x_a)}{w_a}, t_y = \frac{(y - y_a)}{h_a}, \\ t_w &= log(w/w_a), t_h = log(h/h_a), \\ t_x^\star &= \frac{(x^\star - x_a)}{w_a}, t_y^\star = \frac{(y^\star - y_a)}{h_a}, \\ t_w^\star &= log(w/w_a^\star), t_h^\star = log(h^\star/h_a), \end{aligned}$$

where x, y, w, and h denote the boxs center coordinates and its width and height. x, x_a, and $x^\star$ are for the predicted box, anchor box, and ground truth box respectively (likewise for y, w, h).

As mentioned above, the RPN can be trained end-to-end by backpropagation and stochastic gradient descent (SGD). In training the model, the image-centric sampling strategy was employed to ensure each mini-batch arises from a single image that contains many positive and negative example anchors. To optimize the anchors, 256 anchors were randomly sampled in an image to compute the loss function of a mini-batch, where the sampled positive and negative anchors have a ratio of up to 1:1.

3.2.2 Sharing Features for RPN and Fast R-CNN

The last key part of the model is to share convolutional layers between the RPN and the Fast R-CNN. A 4-step alternating training is adopted in Faster R-CNN.

1. Initialize RPN with an ImageNet pre-trained model and train it end-to-end for the region proposal

2. Train a separate detection network by Fast R-CNN using the proposals generated by the RPN.

3. Use the detector network to initialize RPN training, while fixing the shared convolutional layers and only fine-tune the layers unique to RPN

4. Fine-tune the unique layers of Fast R-CNN while keeping the shared convolutional layers fixed

In our case, we adopted the ResNet-50 , ResNext-101-32d, and ResNext-101-64d pre-trained model to initialize the weights.

3.3. Fashion Recommendation

After training the model, we extracted the feature map convolutional layers in Figure 1 of the images. We max-pool and flatten the layers as the deep features. To find the most similar clothes as in the query image, we run a k-nearest neighbor algorithm on those features.

4. Datasets and Preprocessing

The dataset we are using is the Large-scale Fashion (DeepFashion) Database[15] collected by Liu et al. The dataset contains over 800,000 diverse fashion images ranging from well-posed shop images to unconstrained consumer photos. Each image in this dataset is labeled with 50 categories, 1,000 descriptive attributes, bounding box and clothing landmarks.

We only adopt the upper body clothes images to train our model due to the limitation of computation resources. We re-pick the images and combine some categories to have evenly distributed labels, as shown in Table 1. For new classes 0, 1, 2, and 3, all images with the corresponding old labels are included in the new dataset. For new classes 4 and 5, we randomly choose 15000 images in each category, due to the too many images in these categories. After the re-picking, we have 83745 images in total. A random 7500 of them are used as validation set and the others are used as train set.

Table 1. Re-label images to obtain evenly distributed labels

Category	Old label	New label	Total
Anorak, bomber, jacket, parka	1, 4, 11, 13	0	11612
Cardigan	6	1	13311
Sweater	16	2	13123
Tank	17	3	15429
Blouse	3	4	15000
Tee	18	5	15000

To make our images compatible with the detectron toolkit [1], we transfer the images to the COCO style. Examples of images in the dataset are shown in Figure 3.

[1]https://github.com/facebookresearch/Detectron

Figure 3. Image examples of each category in the dataset

5. Experiments and Results

In this section, we will present our experimental settings, results, and discussions. We will show that our model achieves satisfying performance without much hyper-parameter tuning, and it is able to successfully capture the fabric, design, pattern and color of clothes to provide a successful recommendation.

5.1. Experimental Setup

We adopted the settings of [17] as our default setting. We find three ImageNet pre-trained models as our initialization: ResNet-50 (R-50), ResNext-101-32d (X101-32, each branch has 32 channels), and ResNext-101-64d (X101-64, each branch has 32 channels). We train all models for 23k iterations with a batch size of 256. The initial learning rate of the ResNet-50 model is 0.01 and that of the ResNext Models is 0.0025. Learning rate decays by 10 times after training for 12k iterations. A 10^{-4} weight decay is added to all parameters. The pipeline is implemented in caffe2. We planned to tune the hyper-parameters such as weight decay, learning rate, the optimizer adopted, etc., and present their effects on the performance of the model. However, our model reaches very satisfying perfomrance using the default settings, and we confirm that Faster R-CNN is highly generalizable.

In evaluating the model to retrieve image features, we reshape all images to $[64, 64, 3]$ before feeding them into the well-trained Faster R-CNN. We adopt the flattened layer fpn_res5_2_sum_subsampled_2x as the feature layer, and

evaluate the features of all images in the training and validation set. We do the same thing for the query image and run a k-nearest neighbor algorithm on the features to find the best recommendations.

5.2. Quantitative Results

We trained the three models (ResNet-50, ResNext-101-32d, and ResNext-101-64d) for 23k iterations, respectively. Figure 4 and Figure 5 show the validation loss, validation bounding box regression loss, and the validation classification accuracy during training. Since no overfitting was observed with the current weight decay, training loss is similar to validation loss. We only show the curves of the validation set for simplicity.

We may observe that the model achieves amazingly high classification accuracy quickly after 5k iterations. Comparing to the accuracy of our 5-layer CNN baseline, we speculate that the RPN encourages the model to gain better discrimination between the garment and the background. Of course, deepening the network will dramatically increase the hypothesis space and thus help achieve better performance. The regression loss L_{reg} contributes to most of the total loss. We examined the images with bad object detection, and found that most of those images have distortion. We assume the problem can be mitigated by including more real-life images with distortion in the training set.

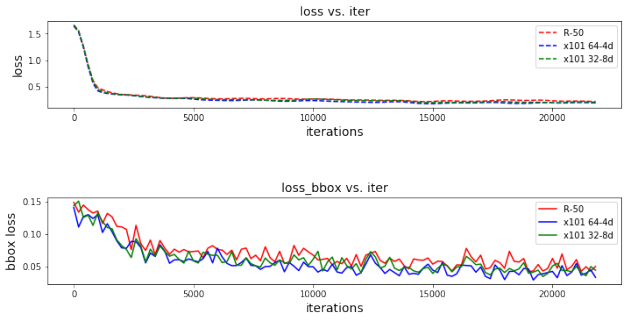

Figure 4. Training curves of the RCNN models

We also compared different models as shown in Figure 6 and Table 2. Faster R-CNN with any CNN model outperforms the baseline in large scale, while ResNext models are slightly better comparing to the ResNet model. However, we did not manage to find the pretrained weights for ResNet-101. Thus it is difficult to assert if the difference comes from deeper networks or the branches in ResNext.

Figure 6 shows the average precision results of each model. We present multiple mAP results, including:

- mAP@[IoU=0.5, area=all]
- mAP@[IoU=0.5:0.95, area=all]
- mAP@[IoU=0.75, area=all]

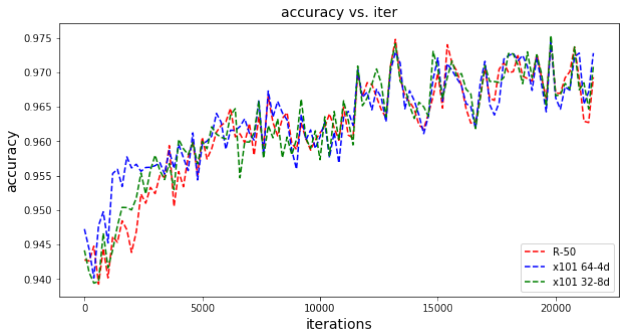

Figure 5. Accuracy curves of the RCNN models

- mAP@[IoU=0.5:0.95, area=small]
- mAP@[IoU=0.5:0.95, area=medium]
- mAP@[IoU=0.5:0.95, area=large]

The AP values are averaged over multiple Intersection over Union (IoU) values and across categories. We make no distinction between AP and mAP. We also evaluate our results on multiple scales of detection objects, "small", "medium" and "large". As we do clothing categorization, we are interested in high average precisions and large areas. We compare these mAPs in the three models, and find that X101-64 and X101-32 models generally have higher APs and the results are very similar between these two. Since detecting the clothing in images is relatively a simple task in object detection (only 1 object among 6 classes), the hypothesis space of any of the three models are enough to achieve a low validation loss. We assume the advantage of the ResNext-101-64d (X101-64) model could be more obvious in more complex tasks.

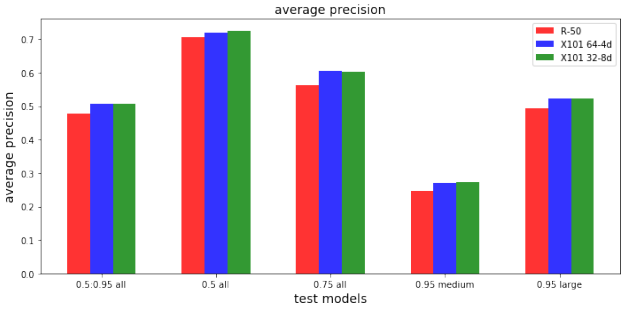

Figure 6. Average Precision of the RCNN models

5.3. Qualitative Results

We propose two types of qualitative analysis to gain more insights into our model: 1) We present the results of object detection on both images in the DeepFashion Dataset

Model	loss	accuracy	mAP@0.5	mAP@[0.5, 0.95]
5-layer CNN	-	0.4922	27.5	15.7
R-50	0.0836	0.9873	70.6	47.8
X101-64	0.0721	0.9922	72.1	50.7
X101-32	0.0702	0.9902	72.5	50.7

Table 2. The performance of the baseline model and RCNN models on the object detection task

and random images found online. 2) We will show examples of recommendation based on the fashion item in the query image.

Figure 7 shows that our fine-tuned model could successfully localize fashion items in images from the training set and the validation. It also works on random online images (including Alexia's Facebook profile photo!), even though the background varies and the person's body may not face the camera. The model also recognizes tanks, blouses, sweaters, etc. easily.

Figure 7. The predicted bounding box for images in the training set, the testing set, and for random online images. Our model could successfully localize the fashion item despite various backgrounds.

Here comes the most exciting and significant evaluation: recommendation. Given a query image, we first extract the feature maps from our Faster R-CNN model in Figure 1, max pool the convolutional layer and then flatten these feature maps into feature vectors. We compute the distances among these vectors and perform k-Nearest Neighbors to find other clothes that are in the same category and have similar styles. The following figure 8 shows the result of recommendation. We pick the top 3 recommendation results, and we can see that the recommended clothes closely resemble the query image in both categories and styles.

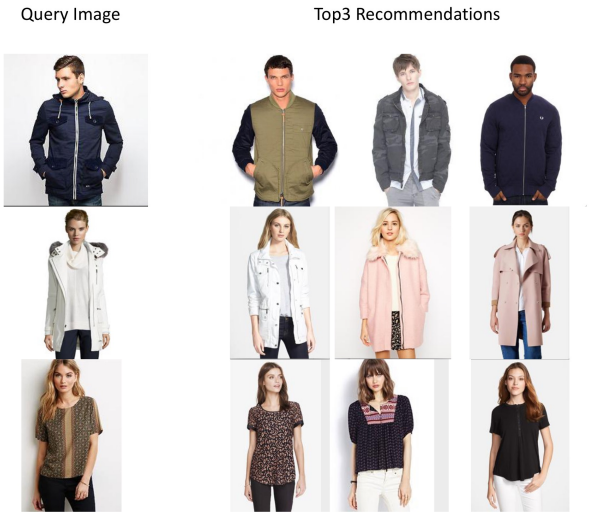

Figure 8. Demo of fashion recommendation based on a query image

Beyond our expectation, we found that the deep features capture not only the category of the fashion item, but also the design, pattern, fabric and color of the query clothing! For the first query image, the model seems to intentionally recommend navy jackets with a zipper (not buttons) for men. For the second query image, our model smartly recommend chesterfield style, medium long, wool coats with light color for women. For the third query, the model recognizes the brownish floral pattern and recommend similar Bohemian style tops. There are much more examples not shown in the report. Overall, we are content with the performance of the image retrieval system.

6. Conclusion and Future Work

In this project, we aim to recommend the most similar fashion items to the one in the query image. We reformulate the task as an object detection problem. By training a Faster R-CNN to classify and localize the clothing, we force the model to not only recognize the clothing in the image but also capture its category and other features. Our model achieves an accuracy of 99% in classifying the clothing items and an $MAP@0.5$ of 0.725 in detecting bounding boxes. Our model was able to detect clothing with high precision in images both from the dataset and from online (including my Facebook profile photo.). Recommending

clothes using the deep features, amazingly, our model is able to recognize the design, fabric, pattern, and color of the clothing! This could be a successful first attempt at the commercialization of fashion retrieval systems.

Due to the limitation of our computing resource, we did not manage to include the lower body clothes and dresses in the training and validation set. We still think the project is a good demo for the image retrieval system. Due to the great performance of Faster R-CNN, we believe the model won't confuse images of tops with that of bottoms and it should only gain better performance after enlarging the training set.

We also plan to develop a smartphone app to gain end-to-end experience of building such a fashion recommendation product.

7. Contributions

All three team members contribute equally to the project. Alexia is responsible for dataset cleaning, the baseline and the recommendation. Fei worked on transfering the dataset to COCO format and trained the model on his GPU. Xiaoyan ran our first tf-faster-rcnn model, but at last we did not use those code. We discuss regularly and debug the code together.

As mentioned in section 5.1, we adopted the detectron code by Facebook AI Research[2] and did not get the chance to tune the hyper-parameters due to the surprisingly great performance of the default setting. We wrote the dataset cleaning, proprecessing, baseline, feature extract and knn ourselves. We wrote 1500 lines of python code in total for this project.

References

[1] L. Bossard, M. Dantone, C. Leistner, C. Wengert, T. Quack, and L. Van Gool. Apparel classification with style. In *Asian conference on computer vision*, pages 321–335. Springer, 2012.

[2] N. Chavali, H. Agrawal, A. Mahendru, and D. Batra. Object-proposal evaluation protocol is' gameable'. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 835–844, 2016.

[3] H. Chen, A. Gallagher, and B. Girod. Describing clothing by semantic attributes. In *European conference on computer vision*, pages 609–623. Springer, 2012.

[4] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005.

[5] R. Girshick. Fast r-cnn. *arXiv preprint arXiv:1504.08083*, 2015.

[6] M. Hadi Kiapour, X. Han, S. Lazebnik, A. C. Berg, and T. L. Berg. Where to buy it: Matching street clothing photos in online shops. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3343–3351, 2015.

[2] https://github.com/facebookresearch/Detectron

[7] J. Hosang, R. Benenson, P. Dollár, and B. Schiele. What makes for effective detection proposals? *IEEE transactions on pattern analysis and machine intelligence*, 38(4):814–830, 2016.

[8] J. Hosang, R. Benenson, and B. Schiele. How good are detection proposals, really? *arXiv preprint arXiv:1406.6962*, 2014.

[9] J. Huang, R. Feris, Q. Chen, and S. Yan. Cross-domain image retrieval with a dual attribute-aware ranking network. In *Computer Vision (ICCV), 2015 IEEE International Conference on*, pages 1062–1070. IEEE, 2015.

[10] S. Jaradat. Deep cross-domain fashion recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*, pages 407–410. ACM, 2017.

[11] W.-C. Kang, C. Fang, Z. Wang, and J. McAuley. Visually-aware fashion recommendation and design with generative image models. *arXiv preprint arXiv:1711.02231*, 2017.

[12] M. H. Kiapour, K. Yamaguchi, A. C. Berg, and T. L. Berg. Hipster wars: Discovering elements of fashion styles. In *European conference on computer vision*, pages 472–488. Springer, 2014.

[13] X. Liang, L. Lin, W. Yang, P. Luo, J. Huang, and S. Yan. Clothes co-parsing via joint image segmentation and labeling with application to clothing retrieval. *IEEE Transactions on Multimedia*, 18(6):1175–1186, 2016.

[14] S. Liu, Z. Song, G. Liu, C. Xu, H. Lu, and S. Yan. Street-to-shop: Cross-scenario clothing retrieval via parts alignment and auxiliary set. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3330–3337. IEEE, 2012.

[15] Z. Liu, P. Luo, S. Qiu, X. Wang, and X. Tang. Deepfashion: Powering robust clothes recognition and retrieval with rich annotations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1096–1104, 2016.

[16] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[17] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[18] E. Simo-Serra, S. Fidler, F. Moreno-Noguer, and R. Urtasun. Neuroaesthetics in fashion: Modeling the perception of fashionability. In *CVPR*, volume 2, page 6, 2015.

[19] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, et al. Going deeper with convolutions. Cvpr, 2015.

[20] A. Veit, B. Kovacs, S. Bell, J. McAuley, K. Bala, and S. Belongie. Learning visual clothing style with heterogeneous dyadic co-occurrences. In *Computer Vision (ICCV), 2015 IEEE International Conference on*, pages 4642–4650. IEEE, 2015.

[21] X. Wang and T. Zhang. Clothes search in consumer photos via color matching and attribute learning. In *Proceedings of the 19th ACM international conference on Multimedia*, pages 1353–1356. ACM, 2011.

[22] T. Xiao, T. Xia, Y. Yang, C. Huang, and X. Wang. Learning from massive noisy labeled data for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2691–2699, 2015.