Discriminative Learning of Apparel Features

Rasmus Rothe[1], Marko Ristin[1], Matthias Dantone[1], and Luc Van Gool[1,2]
[1] Computer Vision Laboratory, D-ITET, ETH Zürich, Switzerland
{rrothe,ristin,mdantone,vangool}@vision.ee.ethz.ch
[2] ESAT - PSI / IBBT, K.U. Leuven, Belgium
luc.vangool@esat.kuleuven.be

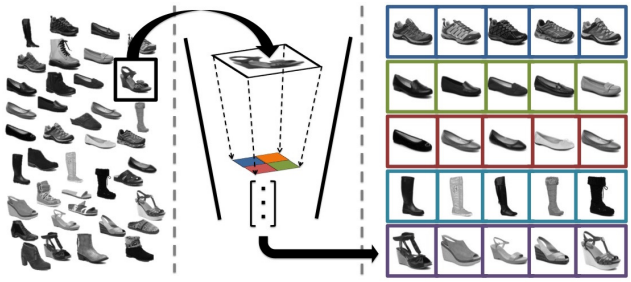
Abstract

Fashion is a major segment in e-commerce with growing importance and a steadily increasing number of products. Since manual annotation of apparel items is very tedious, the product databases need to be organized automatically, e.g. by image classification. Common image classification approaches are based on features engineered for general purposes which perform poorly on specific images of apparel. We therefore propose to learn discriminative features based on a small set of annotated images. We experimentally evaluate our method on a dataset with 30,000 images containing apparel items, and compare it to other engineered and learned sets of features. The classification accuracy of our features is significantly superior to designed HOG and SIFT features (43.7% and 16.1% relative improvement, respectively). Our method allows for fast feature extraction and training, is easy to implement and, unlike deep convolutional networks, does not require powerful dedicated hardware.

1 Introduction

The world-wide e-commerce market for fashion is growing bigger and bigger. Online sales of fashion products are increasing faster than of any other product segment. Total e-commerce sales in the US are expected to surpass $50B in 2014 at 15% anual growth[1].

The product offering of big retailers such as Zappos, Nordstrom, Amazon and Zalando is increasing dramatically. Amazon US has currently more than 500k fashion products in stock and on average 12k new products every weekday [2]. In order to provide the customer with a great user experierence all these new items need to be categorized and augmented with metadata like product category (dress, pants, *etc.*), attributes (material, color, *etc.*) and sub-product categories (pumps, lace-up heels, wedges, *etc.*). This then enables the customer to conveniently browse the product database in an organized manner and retrieve the relevant information easily. To organize a product database in this way, products can be manually categorized and annotated with meta-data, but such endeavour becomes time-consuming and impractical as the database grows. As a result of the very high level of product innovation and competition in the apparel market, categories are subdivided in subcategories, new categories emerge, *etc.* However, since manual re-categorization of products is excessively tedious, better sources of information to cope with such a dynamic setting are required. One such source are the product images, which are often readily available or can be quickly and cheaply acquired. Instead of manual annotation, the meta-data is extracted automatically from the images.



1. Input Images 2. Feature Learning 3. Categorization

Figure 1: Categorization of apparel items using image classification.

Among various approaches to extract the information from images, automatic image classification has shown promising results, and performs well even on large-scale datasets with millions of images and hundreds of classes [1]. Given a set of image categories and an annotated training dataset, a classification model is learned to predict the category of an image unseen during the training of the model. If we need to embed a new fashion product into the existing database ontology, we can predict its precise category by its image. When the product ontology changes, we only need to re-annotate a training set, which is only a small portion of the total data, re-train the classification model and apply it on all the images to re-categorize the products. The accuracy of classification models highly depends on features used to represent the images [10]. The image is usually described with features at different levels of detail. Low-level features are used to describe small patches, while mid-level ones capture larger image regions. Manually engineered low-level features based on gradients such as histogram of oriented gradients (HOG [3]) or scale-invariant feature transform (SIFT [4]) proved to be robust to lighting changes and successfully capture shape cues [2]. This allows to cluster the patches into a vocabulary of so-called visual words. Histograms over different regions are concatenated together to form a pyramid histogram of visual words (PHOG [5]) as a final representation. Vocabulary can be built in unsupervised manner by k-means [5], hierarchical k-means [6] or by iteratively mining discriminative patches [8]. However, when a computational overhead is acceptable, vocabularies can also be learned per class in a supervised manner [9]. These approaches thus employ pre-designed, general low-level features, but learn a specific mid-level representation suited for a particular application.

The discriminative low-level and mid-level features can also be learned jointly by training convolutional neural network operating on raw pixel values

[1] eMarketer 2014
[2] Amazon Associates

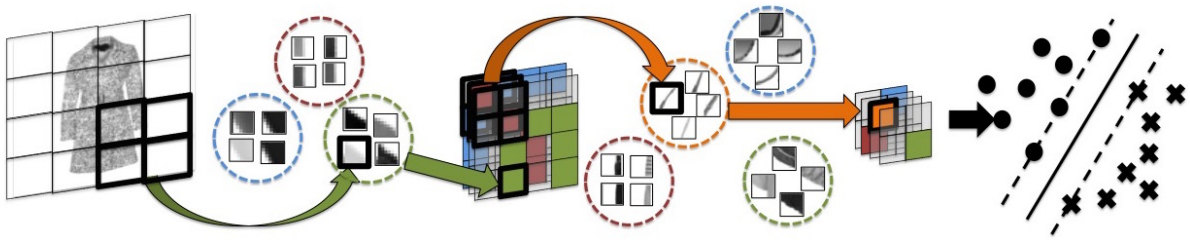


Figure 2: Starting at the raw pixels values, features are learned hierarchically. A linear classifier is then learned on top of the highest-level feature map.

(CNN [10]). While CNNs achieve extraordinary results [1], training is computationally intensive, hard to parallelize, slow and needs specialized hardware.

Due to the importance of the market for online shopping of dressed clothing, various methods have been recently proposed to tackle the problem of classification of apparel images [11, 12, 13, 14, 15]. Most of them work with images taken in unconstrained environments and try to classify the apparel [11, 13], retrieve related images in the database [12] or infer how much a person displays a certain style [14].

Since retailers and shops exchange images taken in constrained conditions, tackling many of the problems associated with a street environment, such as pose estimation [13], induces an unnecessary computational overhead. In contrast, methods like [15] and ours focus on centered, cropped and pre-processed images with uniform background, which are the rule in a business-to-business setting.

Our contribution. The aforementioned methods for classification of apparel images are based on pre-designed low-level features. In contrast, we propose to learn not only the mid-level features, but also the low-level ones. Pre-designed low-level features are engineered for a general purpose, and thus cannot capture the particular discriminative subtleties between the apparel categories. As we show in our experiments presented in Sec. 3, classification accuracy increases when learing low-level features which are custom-tailored for our specific task. Since apparel databases change rapidly, the re-training of the features needs to be efficient. Therefore we can not afford to take weeks to train a CNN [10]. Instead, we propose to learn features at different levels, where outputs from one level are inputs for the next one, in a single pass. At each level, we learn discriminative patches using an efficient k-means clustering approach, starting with the first level that operates on raw pixel values. While the authors of [6] train low- and mid-level features with k-means in an unsupervised manner, we show how k-means clustering can be supervised to achieve better accuracy. On a database of 30k images, we train our hierarchical system (two layers) in less than 2 hours. The accuracy relatively increases by 16.1% when using our low-level features instead of the pre-designed ones.

2 Our Method

Our goal is to learn a set of features for classification of images containing apparel items into a discrete set of pre-defined categories. More specifically, we present a framework to learn those features both in an unsupervised and supervised setting. The features are learned at multiple layers with increasing level of abstraction which is inspired by the architecture of convolutional neural networks [10] and the work of Coates and Ng [6] based on hierarchical k-means clustering.

Initial representation. The features at the lowest level are obtained over the patches of size $s^0 \times s^0$ densely extracted from the given image. A patch is then represented by a concatenation of its grayscale pixel values $x \in \mathbb{R}^{s^0 \cdot s^0}$. We normalize each patch individually to account for differences in brightness and contrast by substracting the mean of its pixel values and dividing by the standard deviation:

$$x'_d = \frac{x_d - \text{mean}(x)}{\sqrt{\text{stddev}(x)^2 + \epsilon}}, \tag{1}$$

where mean(.) and stddev(.) are mean and standard deviation over $(x_1, \ldots, x_{s^0 \cdot s^0})$, respectively, ϵ is a constant to remedy numerical instabilities caused by uniform patches (we set $\epsilon = 10$), and $x' \in \mathbb{R}^{s^0 \cdot s^0}$ the normalized representation of a patch. The initial representation of a patch at level $l = 0$ is then given by $x^0 = x'$.

Higher-level representation. In the following, we explain how to produce a representation at a higher level of abstraction $l+1$ given the features at the level l of a patch of size $s^l \times s^l$. While the initial representation consisted of only a single, grayscale channel, the higher-level features contain multiple, f^l channels. The patch is thus represented as a concatenation of its channel values $x^l \in \mathbb{R}^{s^l \cdot s^l \cdot f^l}$. Provided the codebook $C^l = \left\{ c^l_w \in \mathbb{R}^{s^l \cdot s^l \cdot f^l} \right\}$, we first assign a patch x^l to its nearest codeword w^*

$$w^* = \underset{k \in \{1, \ldots, |C^l|\}}{\arg\min} \ \text{dist}(c^l_w, x^l), \tag{2}$$

where dist(.) is a distance function explained below. The feature map at the level $l+1$ is then obtained by storing a 1 at the w^*-th dimension at the spatial center of patch x^l, and zero otherwise. Thus $f^{l+1} = |C^l|$, *i.e.* the number of channels at the level $l+1$ coincides with the number of clusters on level l. To allow some spatial variations and thereby increase robustness of the features we apply max-pooling [7], *i.e.* a max filter of size $\lambda^l \times \lambda^l$ is applied to each channel separately, and all channels are then downsampled. Note that assigning a patch to the nearest codeword has linear time complexity in $|C^l|$. Small values of $|C^l|$ already yield satisfactory classification accuracy for practical applications, as we show in Sec. 3, and thus the nearest-centroid search is very efficient.

Codebook generation. We generate our codebook C^l based on the patches $\{x^l_i\}$ densely extracted from the training images based on the features at level l. Following the approach described in [6], we cluster the patches using the k-means algorithm. Since standard k-means algorithm often produces correlated clusters, we apply ZCA whitening transform [6] to de-correlate

Table 1: Our method outperforms the hand-engineered features, e.g. 10% higher accuracy compared to SIFT.

	bags	underwear	coats	jewellery	blouses	watches	hats	belts	tops	pants	skirts	jumpers	jumpsuit	dresses	glasses	**avg.**
HOG	57.6	20.4	42.4	35.8	48.4	92.0	69.6	70.8	5.6	81.8	32.6	33.0	46.6	18.8	95.6	**50.1**
SIFT	83.2	39.4	55.4	46.2	35.0	91.4	70.2	79.4	21.2	89.8	58.0	50.6	59.0	56.8	94.8	**62.0**
Ours, unsupervised	78.4	50.6	67.0	61.4	37.8	94.4	77.8	83.6	49.0	90.2	72.4	50.4	72.2	73.0	92.6	**70.1**
Ours, supervised	81.8	53.8	68.8	63.8	42.8	95.6	81.2	88.4	48.8	90.8	74.2	49.6	73.4	72.2	94.2	**72.0**

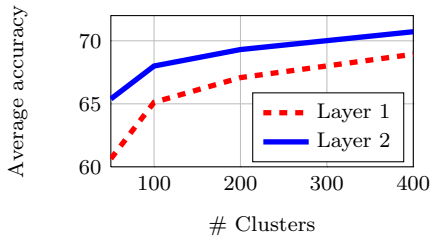


Figure 3: Increasing the codebook size leads to an increase in classification accuracy.

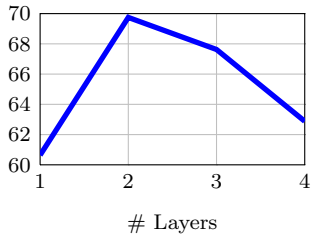


Figure 4: Increasing the number of layers decreases the performances after the second layer.

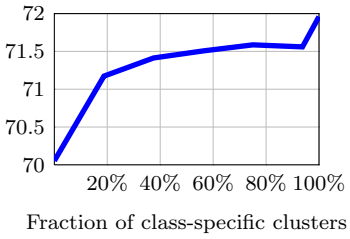


Figure 5: Accuracy increases with supervision. The total number of clusters is fixed to $|C^2|=400$.

the data prior to clustering, which results in more orthogonal cluster centroids and improves the classification performance:

$$VDV^T = \Sigma^l, \qquad x_i'^l = V\sqrt{(D+\epsilon_{\text{zca}}\cdot I)^{-1}}V^T x_i^l. \quad (3)$$

V and D are the eigenvectors and diagonalized eigenvalues of Σ^l, respectively, ϵ_{zca} is a small constant fixed to 0.1 and I is the identity matrix. The cluster centroids $c_w^l \in C^l$ are finally computed based on transformed features $x_i'^l$ instead of x_i^l. Similarly, the distance of a patch represented by x^l to a centroid c_w^l is computed as L2-distance to the transformed features:

$$\text{dist}(c_w^l, x^l) = \|c_w^l - x'^l\|_2. \quad (4)$$

Supervised feature learning. We can leverage class information of an image to learn better features and further improve the classification accuracy. Instead of learning the codebook C^l by clustering all training patches together, we can learn a separate codebook C_κ^l for each class κ individually, and then merge the class codebooks into a single, larger codebook. The union over all the class codebooks $\cup_\kappa C_\kappa^l$ would result in a large number of clusters of which many are not discriminative. Therefore, we use only the subset of these clusters as the final codebook. To pick the most discriminative clusters from a class codebook C_κ^l we first randomly extract a certain number of patches $\{x_i^l\}$ from a separate, validation set of images. Then, we assign these patches to the respective nearest cluster centroid in C_κ^l. For a cluster $c_w^l \in C_\kappa^l$, we compute the fraction of patches assigned to this cluster which are coming from the images of class κ, and use that fraction as a criterion of discriminability. To avoid artefacts clusters with fewer than τ patches are discarded. We select then top N most discriminative clusters of each class codebook C_κ^l, which yields our final codebook.

3 Experiments

We perform experiments on our apparel dataset to assess the performance of our features and compare to pre-engineered features. We show that our unsupervised discriminative features achieve 70.1% average accuracy, compared to 62% by the best-performing pre-engineered features. When we introduce label information in the training of the features at the second layer performance further increases to 72%.

Dataset. The dataset is collected from various fashion e-commerce shops and manually labelled. The majority of the images show the apparel item in front of a clean white background (see Fig. 1). The dataset contains 15 different apparel types (bags, underwear, coats, jewellery, blouses, watches, hats, belts, tops, pants, skirts, jumpers, jumpsuit, dresses, glasses) and counts 30,000 images with 2,000 images per category.

Experimental setup. The dataset is split into 3 parts: for each category we take 1,300 images for training, 200 for validation and 500 for testing. We pad each image with white space to make it square and then rescale it to 128×128 pixels. For our method we fix the patch sizes at different layers for all experiments to $s_0=7$, $s_1=3$, and $s_2=s_3=2$, respectively. Max-pooling is performed with $\lambda_l=2$ between the layers and with $\lambda_l=6$ after the last layer. Clusters with fewer than $\tau=10$ assigned patches were discarded. We observe that slight changes in these parameters do not significantly alter the accuracy. We feed the features concatenated over the whole image at the last layer to one-vs-all linear SVM classifiers for classification. The SVM parameters are optimized on the validation set and fixed throughout all experiments ($C=0.025$). We use average accuracy as performance measure and evaluate the methods on the test set.

Parameters. First, we use no supervision and show how the performance of our features depends on the size of the codebook. We vary $|C^1|$ and measure performance when a single layer is used for classification. Then, we fix $|C^1|=50$ and vary $|C^2|$, while using the second layer as the classifier input. Increasing the number of clusters in both the first and second layer increases performance, *cf.* Fig. 3. As the runtime of the nearest-cluster-mean search increases with larger codebooks, the codebook size is a tradeoff between accuracy and running time. For further experiments, we set $|C^1| = 50$ at the first layer and $|C^{2,3,4}| = 400$ at other layers.

Next we evaluate the impact of the number of layers. Introducing the second layer leads to an increase in 9 points in average accuracy by learning richer features. However, adding more layers decreases the classification accuracy as each cluster lacks training data and and covers too many different visual concepts. That, in turn, makes classification harder, *cf.* Fig. 4. Nonetheless, we hypothesize that increasing both the amount of training data and the number of clusters in the deeper layers might stabilize or even increase performance, as indicated by Coates and Ng [6].

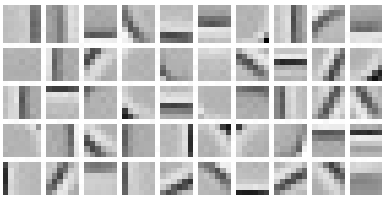


Figure 6: 50 cluster means of codebook in layer 1.

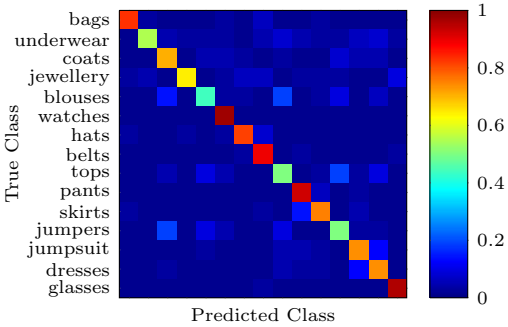


Figure 7: Confusion matrix. Some classes like blouses and tops are more easily confused than others.

Figure 8: The exemplar misclassifications reveal that even for humans correct categorization would be difficult.

We now show how supervision increases average accuracy. In the first layer supervised clustering does improve the performance which is, however, in line with the fact that the patterns in the first layer are not very class specific. To measure the impact of supervision, we train two sets of clusters for the second layer. The first set is trained in an unsupervised manner, while the other one is trained with supervision. We use the union over them as codebook, and fix the total number of clusters to $|C^2| = 400$. To quantify the supervision, we vary the fraction of supervised clusters. The classification benefits already from as little as 20% supervision, and with total supervision the accuracy improves from 70.1% to 72%. Hence, the class information is crucial for learning application-specific features.

Our feature extraction takes 0.1s for the first layer and 0.5s for 2 layers (single-threaded, unoptimized Matlab code).

Comparison to other methods. We compare our features learned with and without supervision against the two other pre-engineered features: SIFT [4] and HOG [3]. We densely extract both of these features over patches of size 16, and create a codebook of 1000 visual words for each of them over the training data. Finally, a histogram of visual words is computed and used as image representation, which we then compare to our representation. Note that our codebook size is smaller, consisting of only 400 visual words, which makes a fair comparison. The results are summarized in Table 1. The average classification accuracy based on hand-crafted HOG and SIFT features achieve average accuracies of 50.1% and 62.0%, respectively, while

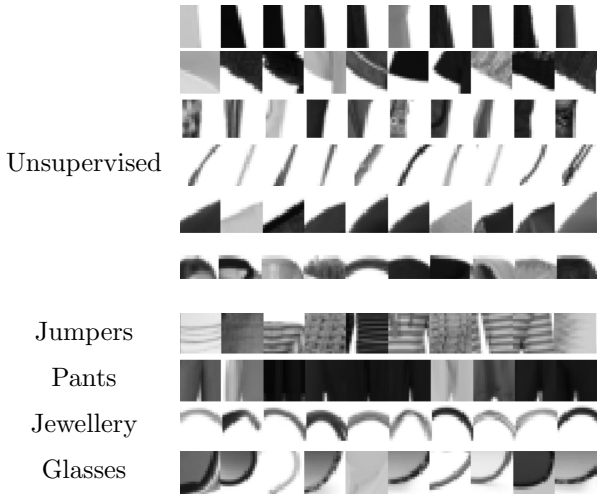


Figure 9: Unsupervised learned codewords in layer 2 show generic shape features while codewords learned under supervision capture very class specific properties.

our method without supervision outperforms them and achieves 70.1%. When we introduce label information in the second layer performance further improves to 72%. Classification of fine-grained classes benefits particularly from our supervised features, *e.g.* for blouses leading to a relative improvement of 13.2%. Thus learning discrimantive features for apparel classification results in a relative improvement compared to HOG and SIFT by 43.7% and 16.1%, respectively.

Qualitative results. Our cluster means in the first layer capture edges at many different directions as well as background and other simple patterns (*cf.* Fig. 6) with strong visual similarity to the features learned by CNNs [10]. While the first layer learns simple shapes, the second layes learns in comparison more complex and composite shapes and textures, *cf.* Fig. 9. Notably, the class-specific clusters capture the characteristical class elements particularly well: *e.g.,* a part of the glasses, a ring or the crotch of pants.

In Fig. 8 we illustrate test images and the classification when our learned features are used. Mind that some of the correctly classified cases are hard to categorize even as a human. The misclassified images reveal that it is hard to distinguish between coats, tops and dresses. The confusion matrix in Fig. 7 shows that certain categories are particularly hard to distinguish from each other, *i.e.* blouses and tops as well as jumpers and coats are frequently mixed up by our method. In contrast most of the other classes can be classified very well.

4 Conclusion

In the light of the fast growth and high level of innovation in fashion e-commerce, automated apparel classification becomes crucial. In this work we proposed a hierarchical discriminative feature learning framework for apparel classification. We show that learning apparel specific features results in better average accuracy and outperforms hand-engineered features, while supervision further boosts the performance. In future work, we plan to refine the clusters iteratively across the layers to improve discriminativeness of the codebook. We believe that our method could highly benefit from feature encoding, *i.e.* Fisher encoding [17].

References

[1] J. Deng, et al.: "ImageNet Large Scale Visual Recognition Challenge," *arXiv:1409.0575*, 2014.

[2] M. Everingham, et al.: "The PASCAL Visual Object Classes (VOC) Challenge," *IJCV*, 2010.

[3] N. Dalal and B. Triggs: "Histograms of oriented gradients for human detection," *CVPR*, 2005.

[4] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *IJCV*, 2004.

[5] S. Lazebnik, et al.: "Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories," *CVPR*, 2006.

[6] A. Coates and A. Y. Ng.: "Learning feature representations with k-means," *Neural Networks: Tricks of the Trade*, 2012.

[7] D. Scherer, et al.: "Evaluation of pooling operations in convolutional architectures for object recognition," *ICANN*, 2010.

[8] S. Singh, et al.: "Unsupervised Discovery of Mid-Level Discriminative Patches," *ECCV*, 2012.

[9] C. Doersch, et al.: "What Makes Paris Look like Paris?," *SIGGRAPH*, 2012.

[10] A. Krizhevsky, et al.: "ImageNet Classification with Deep Convolutional Neural Networks," *NIPS*, 2012.

[11] L. Bossard, et al.: "Apparel Classification with Style," *ACCV*, 2012.

[12] S. Liu, et al.: "Street-to-shop: Cross-scenario clothing retrieval via parts alignment and auxiliary set," *CVPR*, 2012.

[13] K. Yamaguchi, et al.: "Paper Doll Parsing: Retrieving Similar Styles to Parse Clothing Items," *ICCV*, 2013.

[14] M. H. Kiapour, et al.: "Hipster Wars: Discovering Elements of Fashion Styles," *ECCV*, 2014.

[15] X. Wang and T. Zhang: "Clothes search in consumer photos via color matching and attribute learning," *ACM MM*, 2011.

[16] B. Hariharan, et al.: "Discriminative Decorrelation for Clustering and Classification," *ECCV*, 2012.

[17] F. Perronnin, et al.: "Improving the fisher kernel for large-scale image classification," *ECCV*, 2010.