LANDMARKS FOR CLOTHING RETRIEVAL

BY

SHUBHAM JAIN

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2019

Urbana, Illinois

Adviser:

Associate Professor Svetlana Lazebnik

ABSTRACT

Clothing Retrieval is a task that is increasingly becoming popular with the rise of online shopping and social media's popularity. We propose to solve the clothing retrieval problem using landmarks based on the clothing type and features surrounding the landmarks to get a more ingrained view of the design. We compare this method with other models most of which use the whole image as inputs and show the superiority of the model which gives importance to the crucial parts of the images. For the blouses sub-set from of the Deep Fashion dataset[1], we get an 16% increase in the accuracy for the top 3, 14% in top 5 and 11% top 10 retrieval results using the keypoints extraction methods combined with whole images compared to whole images as inputs. We also observe that the clothes retrieved are more similar in terms or design as well as high level properties like sleeve sizes, folded v/s non-folded sleeves etc

To my parents and my friends for their love and support.

ACKNOWLEDGMENTS

Thanks to Daniel McKee (Ph.D. Student in Computer Science, University of Illinois) for his guidance during the project.

TABLE OF CONTENTS

CHAPTER 1: INTRODUCTION

With the rise in usage of e-commerce platforms and the steady increase in the fashion industry, online clothes shopping has recently gained a lot of traction. Recently photo sharing has become very popular through social media websites like Instagram, Pinterest, etc. Searching for clothes based on pictures has never been more important. Most of the time, the searches are related to an inherent design of the cloth and people are looking for close, if not the exact same clothing pattern. Also recently big datasets have been introduced with detailed annotation making it a fertile space for deep learning models to be used. Areas include clothing retrieval [2, 3], landmarks detection [2, 4], image classification etc where deep learning methods have shown to work very well and have pushed the accuracies very high.

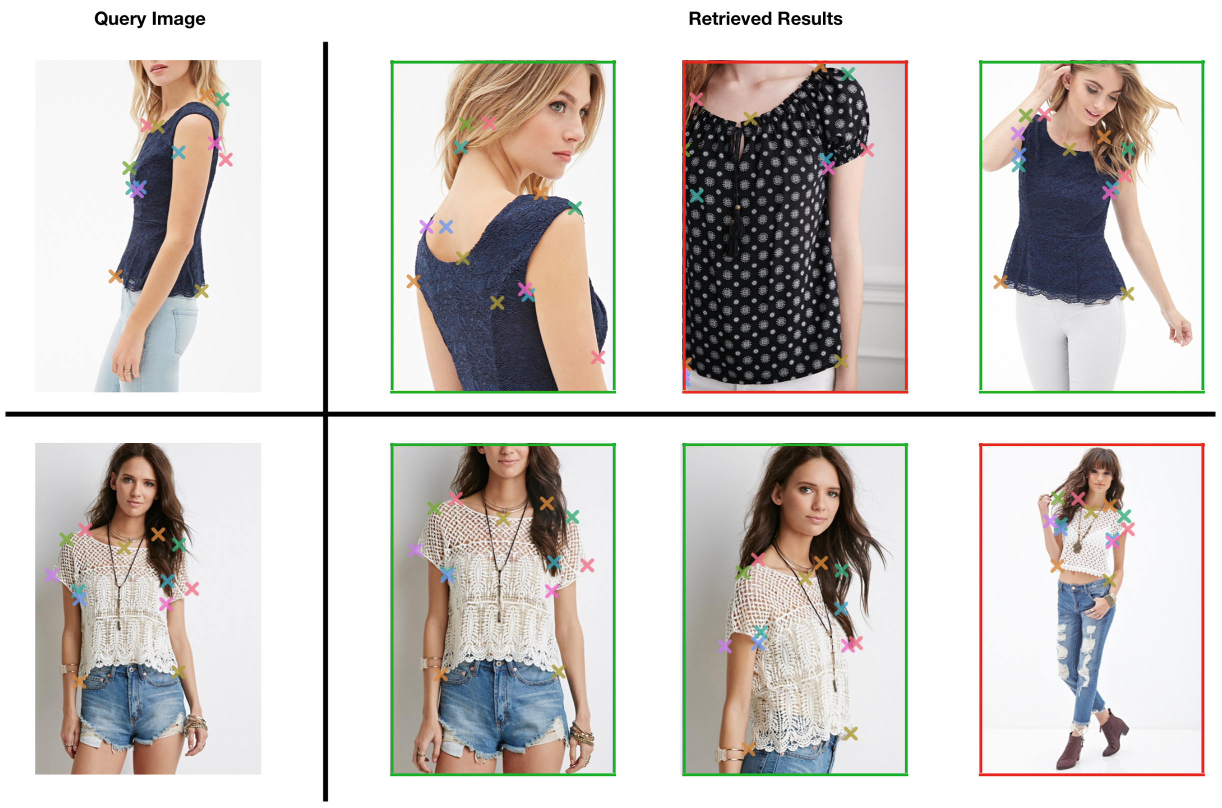


Figure 1.1: Illustration with top-3 retrieved results. On the left we have the query image and on the right we have the retieval results based on the keypoint based feature extraction method. These particular results are from the model in which we extract Zoomout features around the keypoints by passing the query image to a VGG16 network and then using a Cross entropy element-wise product loss (as defined in a later section)

In this work, we will be focusing on clothing retrieval based on landmarks. Previous work in clothing retrieval [3, 5, 6] have shown that neural networks have great potential to solve this problem. Also, landmark detection/localization has been studied very well for areas like face recognition, body pose as well as for clothes. Deep learning methods have been shown to work well for Fashion landmark detection [1, 7] as well. Most of the clothing retrieval methods take the whole image as input and doesn't really focus on the main aspect which is the clothing design. We propose a method which can use landmarks on the cloth as a way to give more focus to the design.

We use Cascaded Pyramid Network [8] for landmark detection and then use Hypercolumns [9] and Zoom-out [10] idea to extract features from a convolutional neural network like FCN [11] or VGG16 [12]. These features are then passed through fully connected layers to get a vector representation of an image (cloth here). We then use either a triplet loss [13] or a modified version of contrastive loss [14] to train the network to learn similarities between the clothes.

Once we learn the feature representation, we train the Nearest Neighbour classifier on our training dataset and query on images from the test dataset and get the top-k retrieval accuracies. Also, we will be using the words landmarks and keypoints interchangeably throughout.

CHAPTER 2: RELEVANT WORK

Visual fashion understanding has recently gained a lot of traction and there have been papers in outfit composition [15, 16] to recognition [17, 2] and retrieval [2, 3]. Before deep learning models were used for these tasks, methods mostly relied on handcrafted features (like SIFT, HOG, etc) [17, 18]. Recently, large scale datasets have enabled to use the deep learning methods for a lot of the tasks and have considerably improved the performance for them.

2.1 FASHION OBJECT DETECTION

The early work in this area was done by Yamaguchi *et al.* [19], where they introduced Fashionista Dataset and used pose estimation and other local features to do segmentation. The same authors then did data-driven learning, essentially a semantic segmentation was learned using nearest neighbours in [20]. The work also used textual context in addition to the image information. Other works include Chen *et al.* [21] proposed a graph-cut based segmentation method to do identity recognition, Lin *et al.* [22] proposed another graph-cut based segmentation method to do upper body segmentation, Hasan *et al.* [23] proposed an approach in which segmentation based approaches was used to assign a label to the clothing item and focused on people dressed up in suits. Wang *et al.* [24] proposed a method for doing clothing segmentation of multiple people utilizing the blocking relationship among people. Recently, Liang *et al.* [25] proposed an RCN network with additional local pixel smoothing, neighbor voting to do pixel level segmentation for clothing.

2.2 CLOTHING CLASSIFICATION

A lot of recent methods do clothing classification in addition to other tasks like landmark localization. Previous work includes Sermanet *et al.* [26] where they did simultaneous human pose estimation and clothing attribute classification based on structured learning. Yang *et al.* [27] proposed a clothing recognition and classification method in surveillance settings. They essentially obtained the foreground segmentation and use that to classify upper and lower bodies to their corresponding fashion items.

Recent methods like Liu *et al.* [2] uses a modified VGG network and split it into 3 parts where 1 part does landmark and the other two do category and attribute prediction.

They also introduced the FashionNet Dataset. Inoue *et al.* [28] augmented and extended the Fashion144k dataset [29] to a weakly labeled dataset. They did unsupervised learning to learn from the weak labels (they did human labeling on a subset of the dataset). Hara *et al.* [30] used a bounding box with R-CNN network for apparel classification and Corbiere *et al.* [31] used their own dataset with text labels to predict those label and learn the representation with negative sampling.

2.3 VISUAL RETRIEVAL

Older retrieval work performed garment retrieval using parsing Yamaguchi *et al.* [19], Di *et al.* [32] where they used global or fine-grained attribute prediction. Hadi *et al.* [3] introduced a new dataset for exact clothing matching and they proposed a method based on pre-trained VGG network and use cross-entropy loss to do the retrieval. Huang *et al.* [33] collected online (when buying) and offline (when trying) data of clothes and used a Siamese network on top to get the best cloth matching so as to get a street to shop setting. Hsiao *et al.* [34] used Latent Dirichlet Allocation (LDA) to find fine-tuned details in images and do retrieval based on that. Yang *et al.* [5] did a more general search for design etc based on any products they may have on the website which has a similar design. Shankar *et al.* [6] augmented the VGG network with another shallow network and focuses more on building a scalable searching model to be applied to their platform.

2.4 OUTFIT COMPOSITION

Work by Mcauley *et al.* [15] takes in a query image (a clothing item) and finds the compositions of accessory, pants, and shirt, etc that go with it and introduces a large scale data with such annotations as well. Li *et al.* [16] used meta-data of fashion items to add more information to the composition modeling and propose to learn fuse modalities. Han *et al.* [35] proposed a bidirectional LSTM to jointly learn a visual-semantic embedding and the compatibility relationships among fashion items in an end-to-end fashion. In addition to suggesting components that go together, they also work and add the task of generating an outfit with multi-modal (images/text) specifications from a user. Nakamura *et al.* [36] proposed a method based on Bidirectional-LSTMs and Auto-encoder to do unsupervised style extraction and generate fashionable outfits according to various preferences like missing component or controlling styles for the same clothing etc.

2.5 FASHION EMBEDDINGS AND FORECASTING

Simo *et al.* [37] used a Siamese network to do weak label learning based on Feature CNN and Softmax loss. Vasileva *et al.* [38] proposed a method to add text-semantic info in addition to the visual information to learn and embedding space much more distant to give space to a lot of different kind of queries. Al *et al.* [39] talked about different things that can be represented as fashion and how we can predict how much they get sold over time.

2.6 IMAGE GENERATION & STYLE TRANSFER

Zhu *et al.* [40] used a Generative Adversarial Network (GAN) to modify clothes in an image based on a textual description. Raj *et al.* [41] proposed a 2 stage model based on U-net that can interchange garment appearance between two single view images of people with arbitrary shape and pose.

2.7 LANDMARK DETECTION ON CLOTHES

Landmark detection previously was studied with joint localization in contexts like face alignment [42] and human body pose estimation [43]. Ramanan *et al.* [42] used handcrafted features (which was also a limitation to the model) to model facial landmarks as distinct parts and learned a tree-structured model to find the configuration that suits best. Tompson *et al.* [43] used a multi-scale Fully Convolutional Network (FCN) network to use the idea of sliding window and pyramid feature extraction.

Recent methods have focused on images in the wild instead. Liu *et al.* [1] introduced a dataset called Fashion Landmark Dataset (FLD) which has the labels and landmarks. They also propose a deep fashion alignment (DFA) framework and uses Auto-routing to classify easy and hard to label points. Yan *et al.* [7] introduced a dataset that has detection without landmarks as ground truth. They propose Selective Dilated Convolution for handling scale discrepancies, and a Hierarchical Recurrent Spatial Transformer for handling background clutters inside the network for better classification. Wang *et al.* [44] introduced a Bidirectional Convolutional Recurrent Neural Networks (BCRNNs) which has fashion grammar (constraints over landmarks) and attention and does landmark prediction and clothing categorization.

2.8 KEYPOINT FEATURE EXTRACTION

Keypoint extraction has been used to do keypoint matching for a long time using hand-crafted features like SIFT [45] and HOG [46]. New feature extraction methods based on deep learning models were introduced for the same. Hariharan *et al.* [9] introduced hypercolumn find feature representation of a point in the image through a CNN network by extracting the values from different layers for the network corresponding to the same location. Mostajabi *et al.* [10] introduced zoom-out features to get features for a superpixel to segment it to the correct object. They extended the hypercolumn idea in a way by taking mean across the superpixel and get the features across various layers.

CHAPTER 3: METHODOLOGY

In this section, we will describe the various models we used. We then move to the loss functions and then our exact methodology

3.1 VGG 16

VGG was introduced by [12]. It is a Convolutional Neural Network followed by 3 Fully Connected Layers. Figure 3.1 describes the structure of the network. Essentially it has convolution layers, maxpool layers, relu activation and fully connected layers. The model was used to do localization and classification and scored 1_{st} and 2_{nd} for the ImageNet Challenge 2014. They showed how using 3x3 kernel convolution and increasing depth of the network can be effective. They did a thorough evaluation of how depth can affect the performance. The network takes in a 224x224x3 image and passes it through the network more extensively defined in figure 3.2.

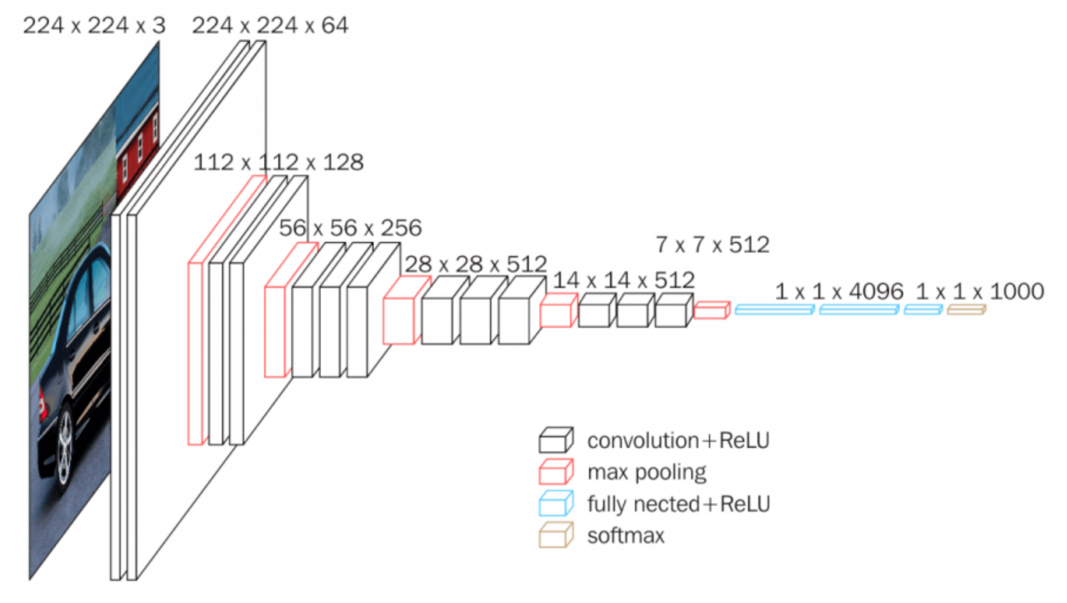


Figure 3.1: VGG 16 model: Image taken from [47]

Essentially it is a combination of 3x3 convolution layers with an increasing number of channels. After every 2-3 convolution layers, maxpool is applied to reduce the shape of the intermediary result. The output of the final maxpool layer is then followed by 3 fully connected layers which reduce the side to 4096. So, we can represent an image with a 4098 vector which is the output of the model.

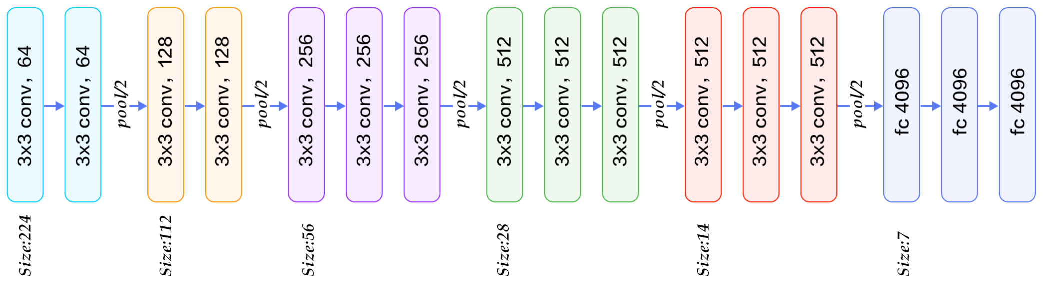


Figure 3.2: VGG 16 layers: Image taken from [48]

3.2 FULLY CONVOLUTION NETWORK

This network was used by [11] for doing images segmentation. It is essentially an encoder-decoder network with convolution, maxpool, and unpooling layers. The network is illustrated in figure 3.3. The encoder part is where the input is reduced to it's smallest representation which is before the conv6 part in the image. The decoder part is followed after that when the input is generally mapped back to its original shape. The network is similar to VGG in the sense that we will be using 3x3 convolutions and the encoder part is the VGG network followed by layers to increase the size to the original shape.

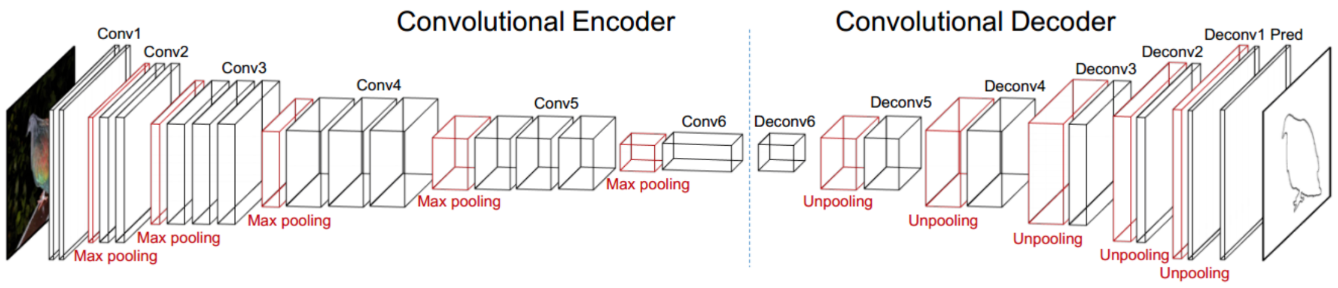


Figure 3.3: FCN network: Image taken from [49]

3.3 K-NEAREST NEIGHBOUR CLASSIFIER

K nearest neighbour classifier is a classifier based on nearest neighbour voting. Essentially a vector is classified based on the labels of its ”k” nearest neighbour. We generally find the best k and also a good distance mapping for the vectors to get the best results. In the example in figure 3.4 let say the green circle is one vector we need to assign a label for. If k was 3 we see that we have 2 red triangles in the 3 nearest neighbour and hence we will assign the green circle as a red triangle. But if k was 5, we can see that there are 3 blue rectangles in the 5 nearest neighbours and hence the label then would be a blue rectangle. Another aspect of the nearest neighbour classifier is how to find the nearest neighbour from a time complexity point of view and there are methods like Kd-tree etc which are faster ways to go about it.

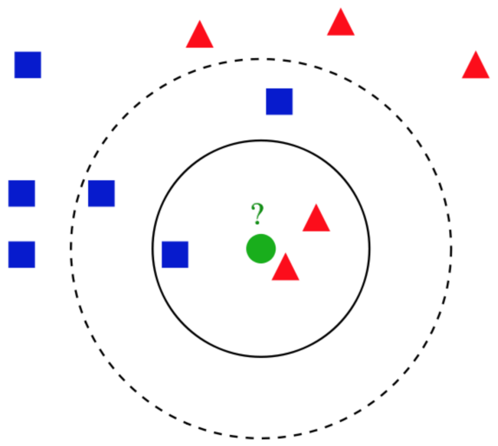


Figure 3.4: K-NN example: Image taken from [50]

3.4 HYPERCOLUMNS

Hypercolumns was introduced by [9] as a way to extracting features from an image. Feature extracted from the network like VGG is too coarse for certain tasks like segmentation etc and hence the need for a more semantic representation with localization as well. The paper showed its effectiveness in three tasks: simultaneous detection and segmentation, keypoint localization and part labeling. The idea to extract the values from a certain location from all the layers as use that a descriptor for the certain location. This will give a wider representation because of the field of reception of the convolutions (and pooling layers if any). The idea is illustrated in figure 3.5. The paper also talks about how to use hypercolumns differently for these tasks for example for segmentation asks they use hypercolumns for refinement rather than replacing that with the whole network.

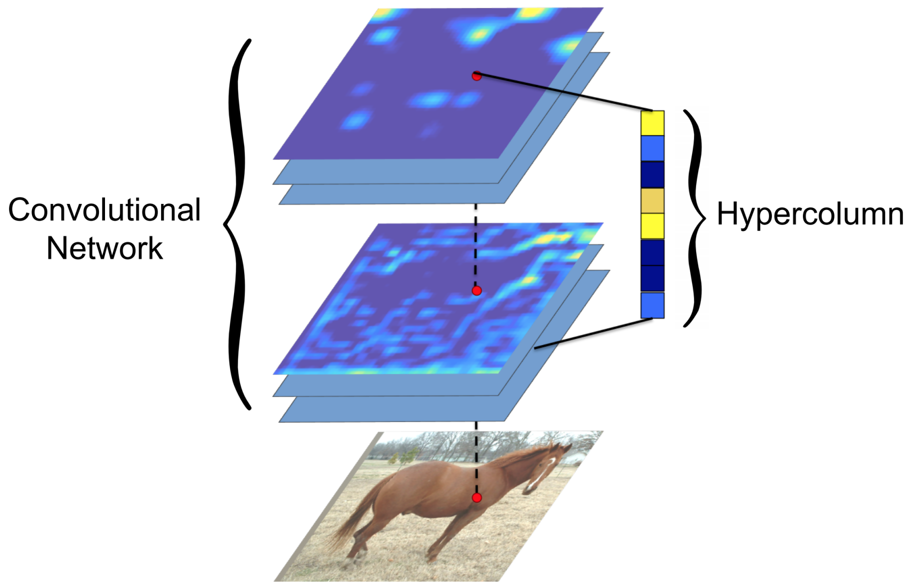


Figure 3.5: Hypercolumn example: Image taken from [9]

Essentially, we are taking the features from different layers for that location and we will get low-level features from the initial layers and high-level features from the final layers. This in a way gives a more holistic representation of that location.

3.5 ZOOM-OUT FEATURES

Zoomout features were proposed in [10] where they introduced a purely feed-forward architecture for semantic segmentation. The idea is illustrated in figure 3.6. Essentially we want to get a feature representation of a superpixel using some model (in this diagram it is similar to the VGG network). This is different from hypercolumn because there we were looking for feature representation for a particular location (pixel) on an image. The essential idea is to extract the features from the layers corresponding to the superpixel location and mean them over for a particular channel leading to a vector of size of the number of channels from that particular layer and concatenating all such vectors to get a feature representation of the superpixel which has low level features (from the lower layers) to high level features (from the higher layers).

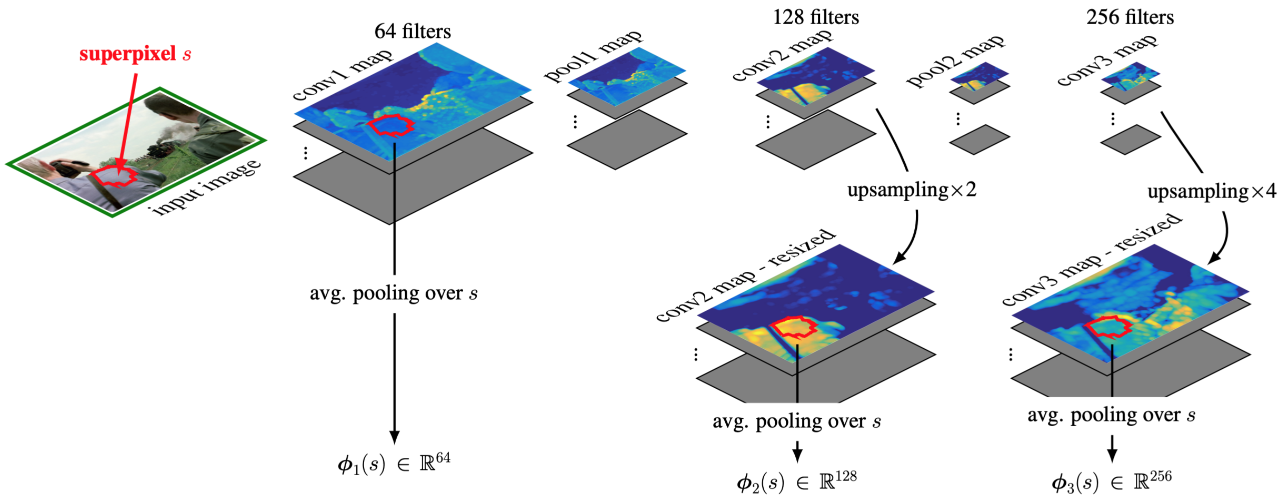


Figure 3.6: Zoomout example: Image taken from [10]

3.6 CASCADED PYRAMID NETWORK

Cascaded Pyramid Network (CPN) was introduced by [8]. The paper used the network on the task of multi-person pose estimation and majorly focused on landmark detection especially the "hard" landmarks which may be occluded etc. The network includes two sub-networks: GlobalNet and RefineNet. GlobalNet is based on a pyramid network and can localize simple landmarks like eyes and hands but may not be good for precisely localizing the difficult landmarks which may be occluded or not visible. The second sub-network RefineNet tries to explicitly handle the hard landmarks by combining different levels of feature representations that were obtained from the GlobalNet and uses an online hard keypoint mining loss.

The online hard keypoint mining loss is essentially an L2 loss on keypoints that are harder to detect. The way these points are detected is via training as the "simple" keypoint will have a smaller L2 loss and hence they will get a lot of importance. To maintain the balance so that the training also focuses on the not so simple keypoints, the RefineNet uses the L2 loss only on the landmarks that have higher L2 loss. The network is illustrated in figure 3.7.

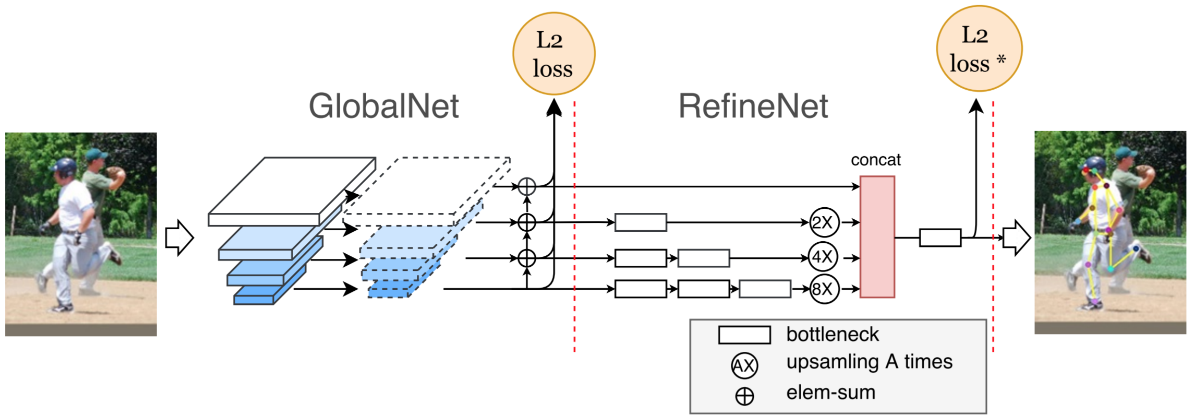


Figure 3.7: CPN training: L2 loss* means L2 loss with online hard keypoints mining. Image taken from [8].

Also, it is worth adding that to work with multiple humans in an image, the network first has a Human Detector which gives bounding boxes to the network around the humans present in the image and that bounding box is the input to the GlobalNet as compared to the whole image itself. Also, the backbone of the network is based on ResNet which is used for both GlobalNet and RefineNet.

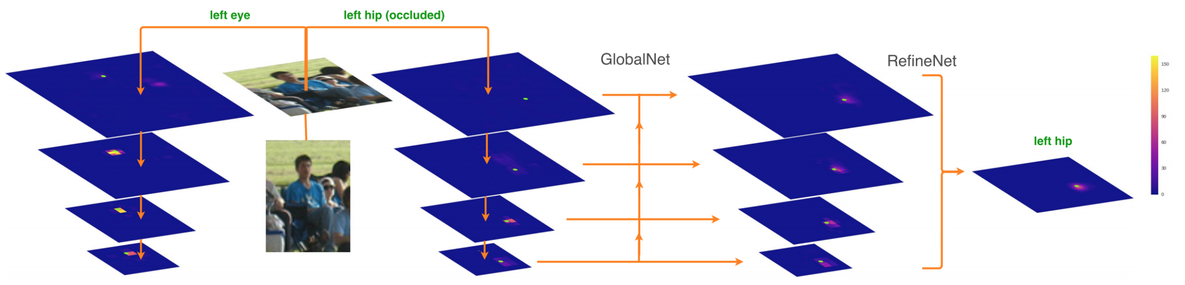


Figure 3.8: CPN output heatmaps: The green dots means the ground-truth location of keypoints. Image taken from [8]

Figure 3.8 shows how the network prediction works for two kinds of keypoints. For a "simple keypoint" only GlobalNet is used whereas for a "difficult" keypoint left hip" is predicted with using both the GlobalNet and the RefineNet.

3.7 CONTRASTIVE LOSS

This loss function was introduced by [14]. The basic idea of this loss is given two vectors (say X_1 and X_2), we first find normalize them and then find their Euclidean distance and call it D (short for $D(X_1, X_2)$). We then use a margin loss function for this distance essentially saying that the positive samples should be close to each other and the negative samples should be away by a certain margin (m). The label (Y) is 1 for positive samples and 0 for negative samples. The equations for this loss looks as follows:

$$\mathrm{D}(X_1, X_2) = ||X_1 - X_2|| \tag{3.1}$$

$$\mathrm{L}(Y, X_1, X_2) = \frac{1}{2}((Y) * (D)^2 + (1 - Y) * (\max(0, m - D))^2) \tag{3.2}$$

3.8 LOGISTIC DOT PRODUCT LOSS

This is a different version of the Contrastive loss where we change the Euclidean distance to dot product and the squaring to log. The idea is to have the two vectors of different labels to be activated in different dimensions of the vectors leading to a smaller dot product. We do normalize the vectors before the dot product. The equations are as follows:

$$\mathrm{D}(X_1, X_2) = X_1.X_2 \tag{3.3}$$

$$\mathrm{L}(Y, X_1, X_2) = -((Y) * \log(D) + (1 - Y) * \log(1 - D)) \tag{3.4}$$

3.9 CROSS ENTROPY ELEMENT-WISE PRODUCT LOSS (CEEP)

In this loss we do a very similar thing as Logistic Dot product but, instead of taking the dot product we take the element-wise multiplication (denoted by $\odot$ and pass that vector to a Fully connected (FC) layer which outputs a 2 length vector and we do a softmax and a negative log-likelihood loss on top of that. So instead of summing up all the dimensions of the vectors, we pass it through an FC layer which can be learned while training with the other components of the network. The equations are as follows:

$$\mathrm{E}(X_1, X_2) = \mathrm{Softmax}(\mathrm{FC}(X_1 \odot X_2)) \tag{3.5}$$

$$L(Y, X_1, X_2) = -((Y) * \log(E) + (1 - Y) * \log(1 - E)) \tag{3.6}$$

3.10 TRIPLET LOSS

Triplet loss was proposed by [13]. From our experiments, the element-wise loss was working well so we thought of incorporating that into the triplet loss function. Essentially we get a set of three vectors (Say X_1, X_2, X_3). Let X_1 be the vector to compare the two vectors with. One vector (say X_1) is more similar to X_1 than another vector (X_3). So we take the element-wise product of X_1 and X_2 and of X_1 and X_3. Pass those through a Fully connected (FC) layer which gives 1 value as the output and do a margin between the value from the first pair and the second pair essentially try to say that the values from the two should differ from a margin amount. Let us call $E(X_1, X_2)$ as E_{12} The equations are as follows:

$$E(X_1, X_2) = \text{Softmax}(\text{FC}(X_1 \odot X_2)) \tag{3.7}$$

$$L(X_1, X_2, X_3) = \min(E_{13} - E_{12} + m, 0) \tag{3.8}$$

Now that we have described the underlying networks and losses we used, we can go ahead and discuss the underlying method we used. We first trained the CPN network on Tianchi dataset to predict landmarks. For example, blouses had 13 landmarks and dress had 15 with some overlaps. We then used the DeepFashion In-shop dataset for the retrieval tasks. We observed that the images from Tianchi and DeepFashion didn't have a stark difference and the model trained on Tianchi worked well on DeepFashion images so we didn't train it on DeepFashion's landmark dataset.

Some of the results landmarks on Tianchi dataset are shown in figure 3.9. These images are taken from Tianchi dataset. Figure 3.10 has images from DeepFashion In-shop dataset and landmarks from the same network (trained on Tianchi images). Even though there were more non-straight poses, the network was able to predict those hard keypoints well even when they were occluded, etc. A lot of images weren't also full length and hence there was variability in the human pose, what portion of the body is visible etc but the network still worked well on DeepFashion even though it was trained on Tianchi dataset.

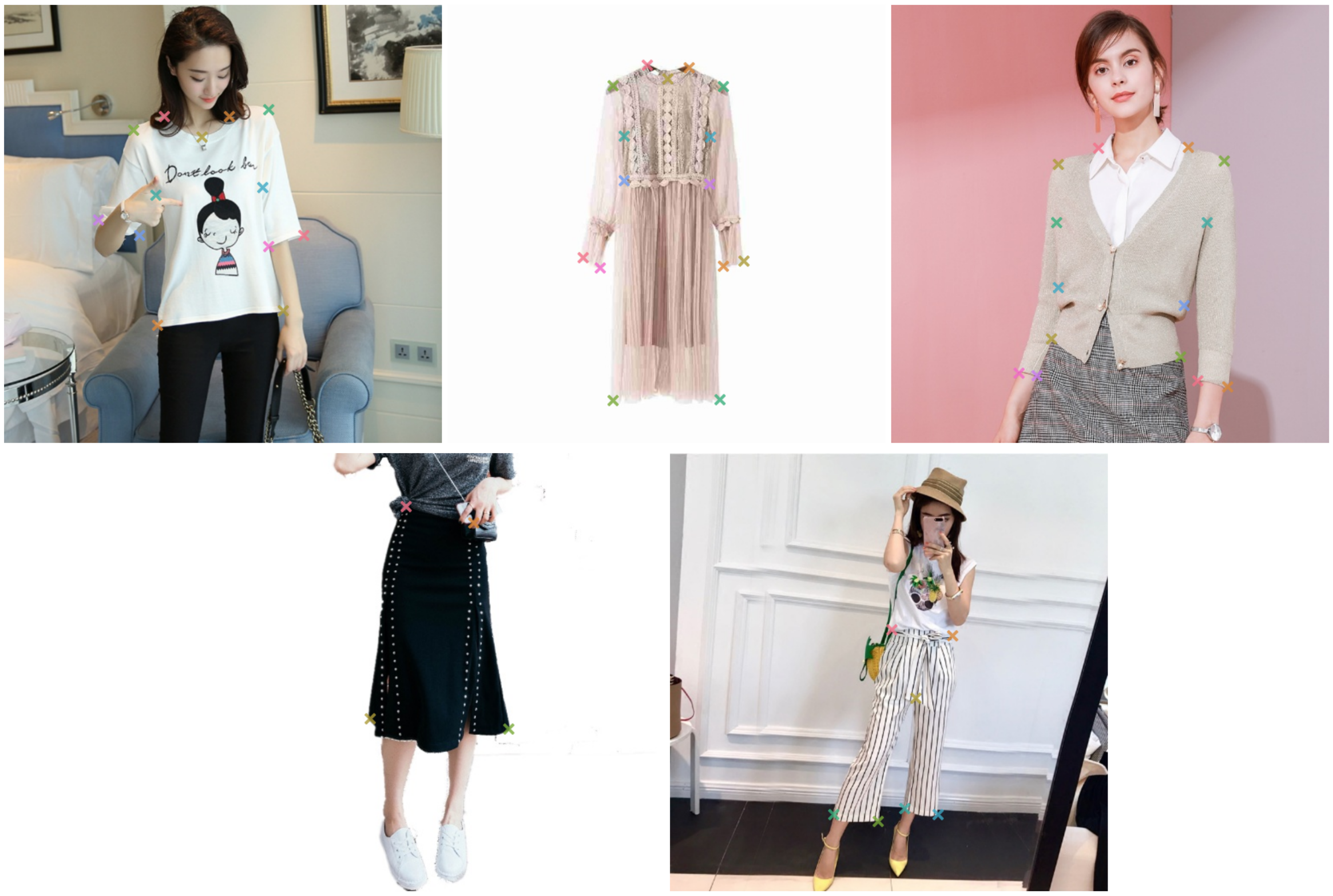


Figure 3.9: Landmark results: CPN network trained using Tianchi dataset on Tianchi images

For every clothing we had, we picked randomly one orientation out of the many for a clothing to be in the query set and rest went to the training set (3-4 images). We then used the keypoints and extracted features around it using the feature extraction methods we described earlier (Zoom-out and hypercolumn). Just concatenating the keypoint patches either side by side or as channels and passing that through a Convolution Neural Network didn't work. We then passed these feature through 3 Fully connected layers and got a d length vector. So with a pair of images and label for similarity, we get two d length vectors which are then passed to the loss function and then the whole network was trained.

For the retrieval, we used the training images and pass them through the Convolution Neural Network and the Fully connected layers to get the d length feature vector corresponding to those images. We then trained a K-Nearest Neighbour classifier on these vectors. We then processed the query images (that we set aside before training) by passing them through the same network and getting a d length vector. We then get the label for this image using the K-Nearest Neighbour classifier that we trained. We report the accuracy, top k results and other relevant metrics in the results section.

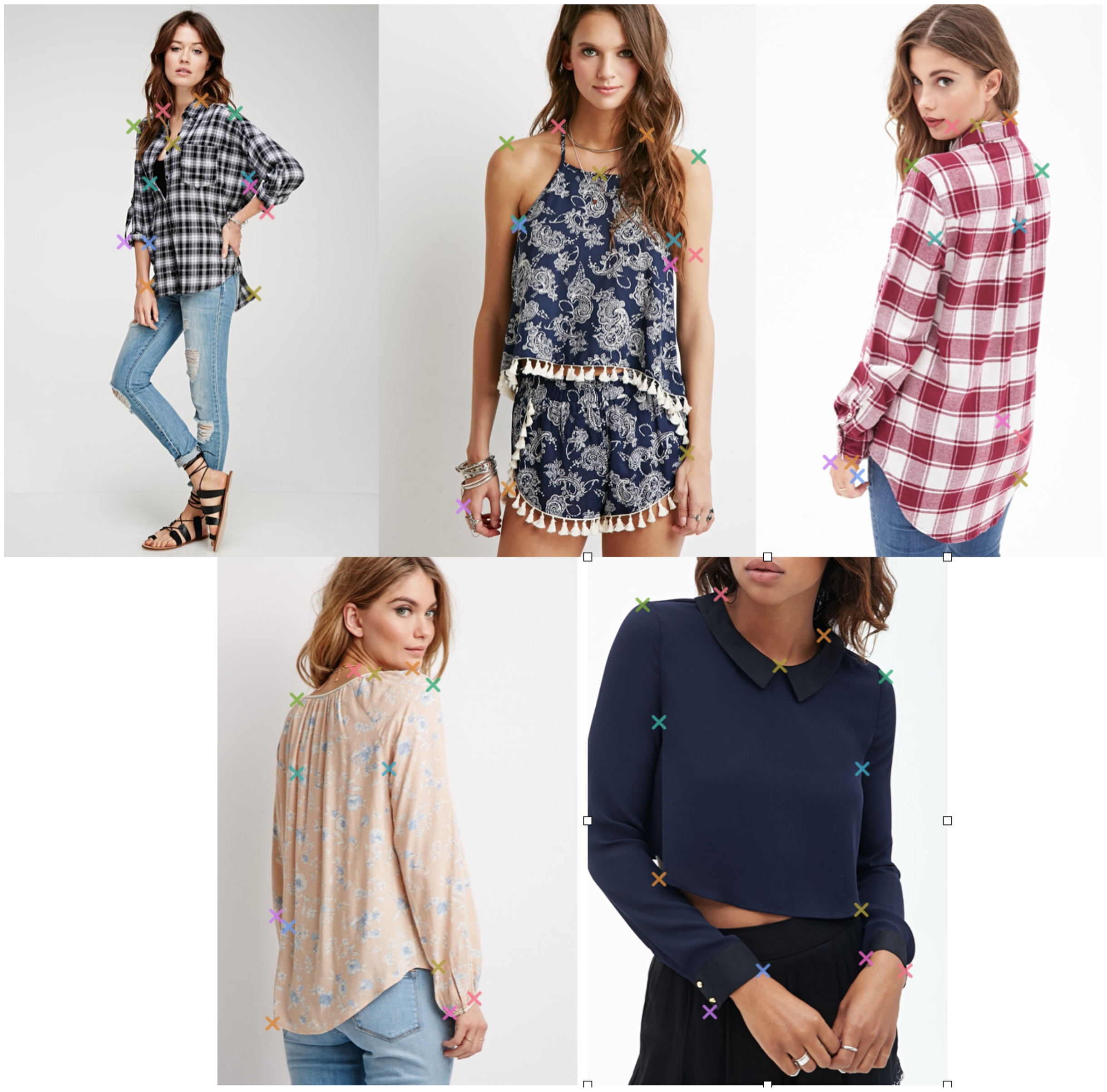


Figure 3.10: Landmark results: CPN network trained using Tianchi dataset on DeepFashion images. Also note the variations in the images

CHAPTER 4: EXPERIMENTS

In this section, we will talk about the two datasets that we used for the different pipelines for the model. One of these datasets was released as a competition and the other as a paper. We will then talk about the experiments we did base on them and the results we achieved.

4.1 TIANCHI COMPETITION

"Keypoints Detection of Apparel - Challenge the Baseline" [4] was the name of the competition from Tianchi lab and was co-hosted by the Alibaba Image and Beauty team in collaboration with the Textile and Clothing Department of the Hong Kong Polytechnic University. The competition had two tasks: clothing key point positioning and apparel attribute label recognition. We used the dataset for the landmarks to train our landmark detection model which we will describe in a later section. The landmarks labeling looked like the following:

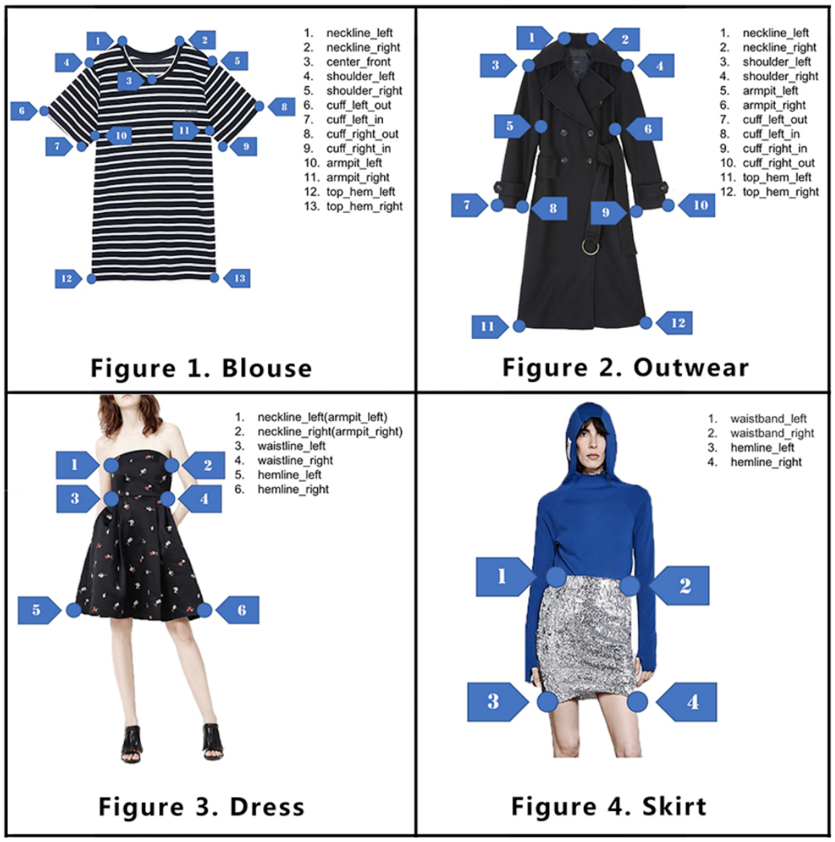


Figure 4.1: Tianchi dataset: Image taken from [4]

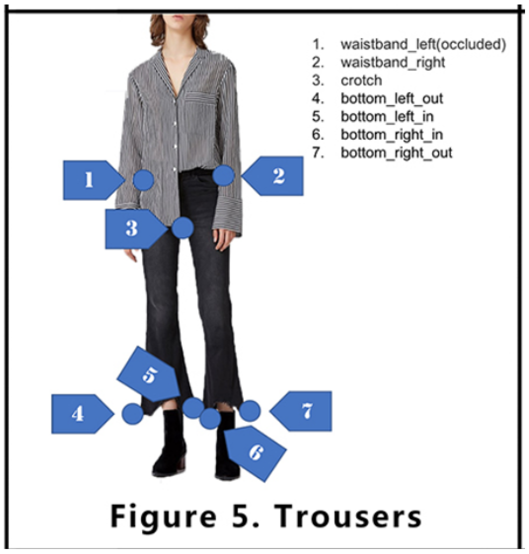


Figure 4.2: Tianchi dataset: Image taken from [4]

Essentially, the dataset had 24 types of keypoints but some of them were non-existent in some of the categories as can be seen in figure 1. We mostly experimented with the blouses dataset for keypoints to test out the models initially. There were specifically 16613 images for blouses with landmark annotation.

4.2 DEEP FASHION

This dataset was released with the [2] paper and has 4 kind of annotations Category and Attribute Prediction, In-shop Clothes Retrieval, Consumer-to-shop Clothes Retrieval, and Fashion Landmark Detection. We used the inshop clothing dataset to do retrieval and has reported most of the results based on it. The inshop dataset is shown in figure 3.3

We had a total of around 8000 blouses images in which every cloth had approximately 4 images of with different poses. So around 1600 classes of blouses. We also had other upper body clothes like Cardigans, Jackets, Coats, etc.

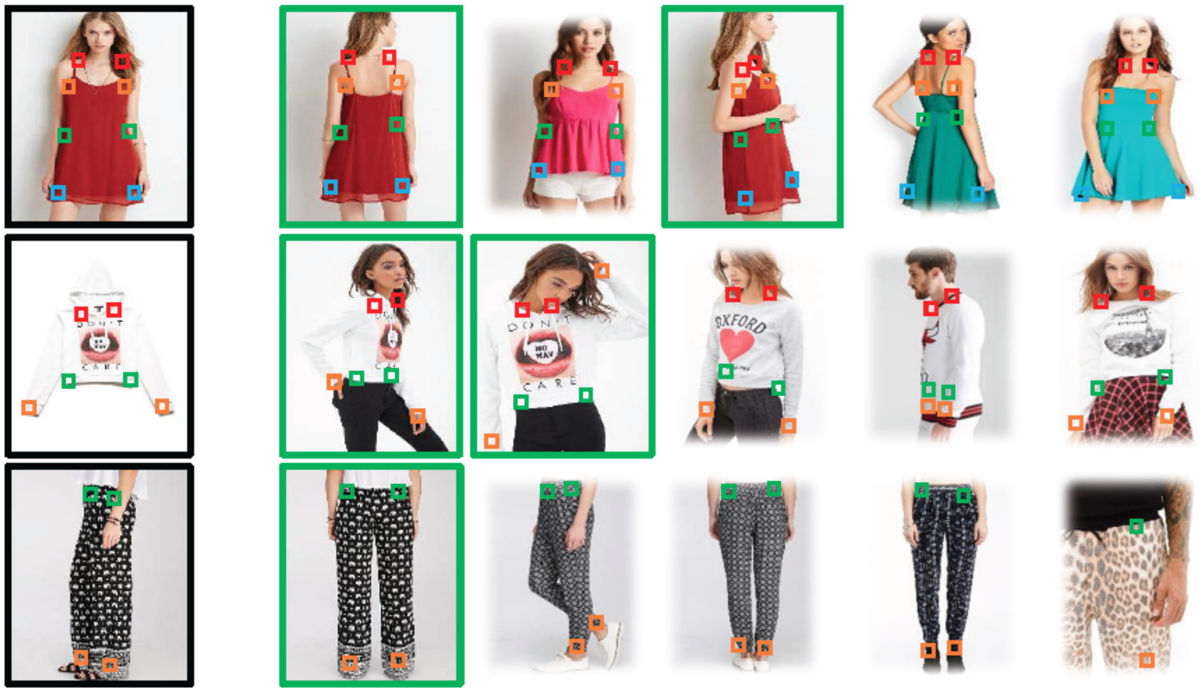


Figure 4.3: Deep Fashion Inshop retrieval dataset: Image taken from [1]

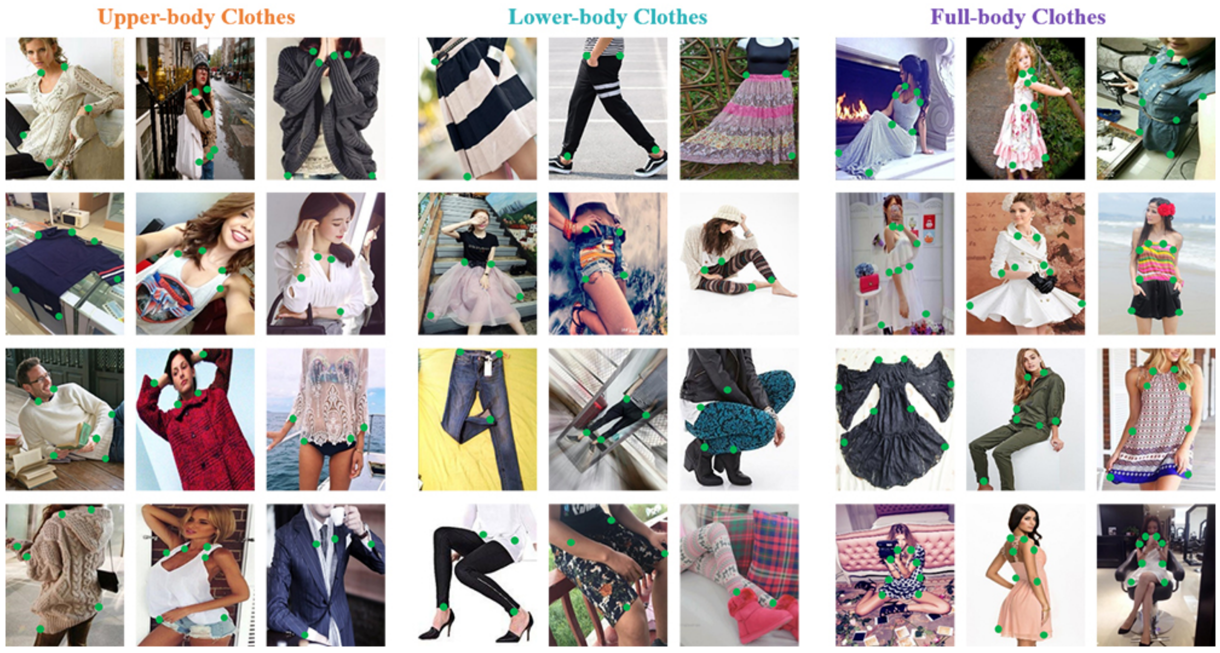


Figure 4.4: Deep Fashion Landmarks Dataset: Image taken from [1]

Since we wanted an already well-working landmark model for fashion dataset, we looked for an implementation on either DeepFashion or any other fashion dataset and that is where we found this https://github.com/gathierry/FashionAI-KeyPointsDetectionOfApparel on GitHub whose model was trained on Tianchi dataset and was also able to handle the upper body, lower body as well as full-length clothes. We then trained this network on 5 categories: blouse, dress, outwear, skirt, trousers on Tianchi dataset and since this dataset didn't have clothing retrieval, we used the DeepFashion dataset for retrieval (which also had landmark annotation on a different subset of its data). Most of the results for landmark localization were good as shown in figure 3.10. Some of the inaccurate result cases are shown in figure 4.5

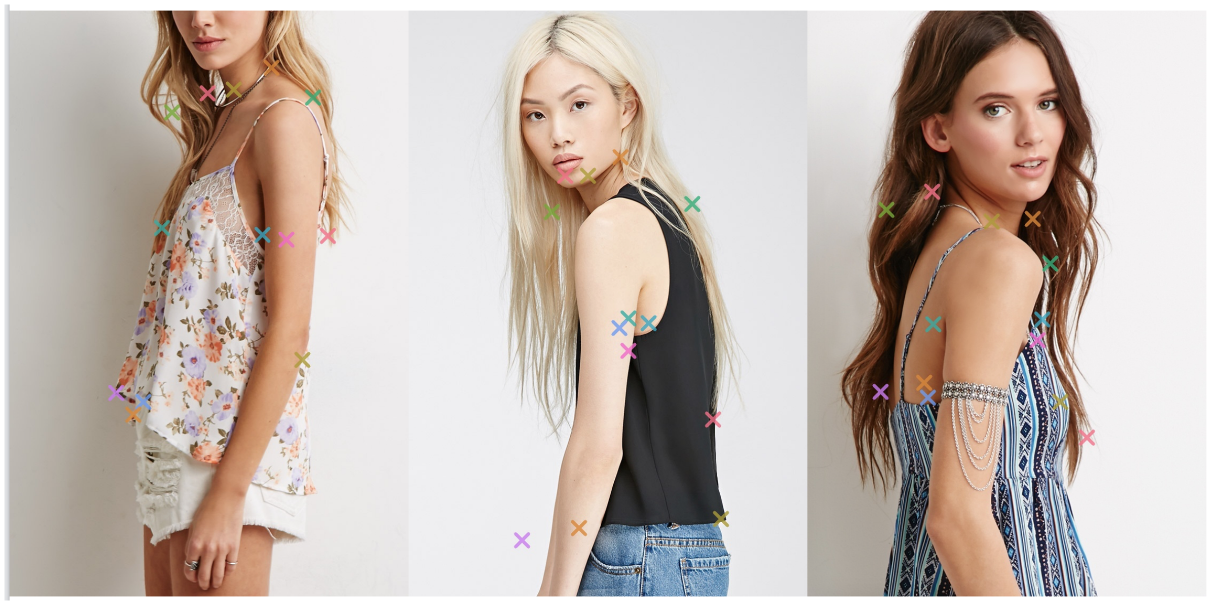


Figure 4.5: Landmark inaccurate results: CPN network trained using Tianchi dataset on DeepFashion images.

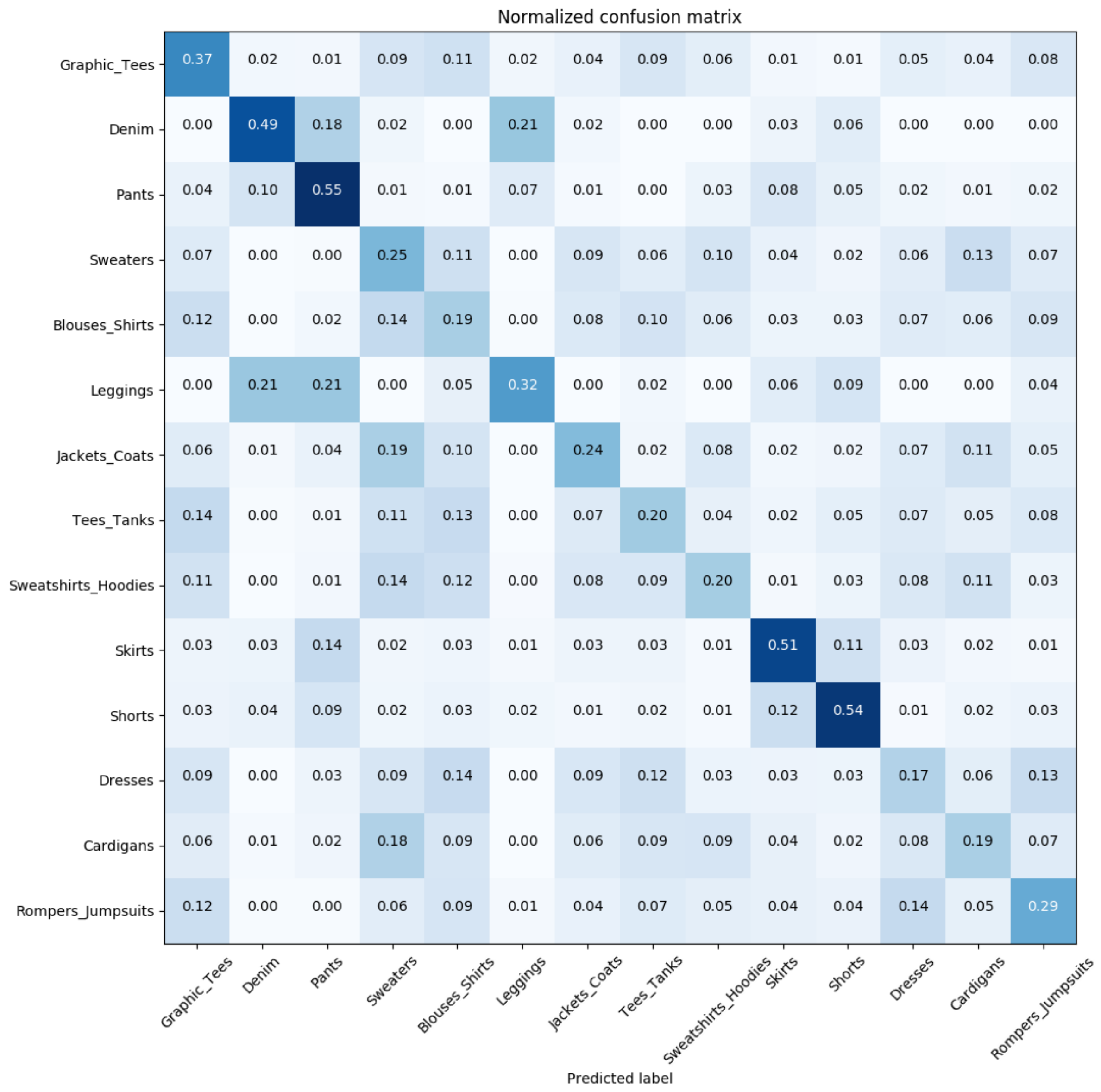
We had an option of either combining different types of clothes (upper body, lower body, etc) from both datasets and make 1 single model predict all the landmarks and then use it to go general clothing retrieval or can work specifically on upper body, lower body at a time. To test the model's capability, we sticked to the later and used the blouses dataset.

We also had original annotation which assigned images as either training or querying in the DeepFashion dataset but we realized there was something wrong as certain folders had too many query images and very few images of the same class in train and hence we decided to define our own query and train annotation. As described in the methodology section, we picked randomly one orientation out of the many for a cloth to be in the query set and rest went to the training set (3-4 images). This gave us around 80% training (around 6k) and around 1.6k query images for the blouses. We also had multiple colors of the same designs and we also considered them as different classes.

We trained the model (VGG16 convolutional layers followed by 3 FC layers) with the input as keypoints patches stacked onto each other (in a row and column manner) with Contrastive Loss (that we defined earlier). This model didn't train well i.e. the loss wasn't decreasing a lot after the first epoch. So, we decided to stack the key point patches as channels (input becomes size [patchsize, patchsize, 3*numkeypoints]) which also didn't train well. We then changed the VGG model to a smaller CNN network thinking maybe the learning is slow because of the depth and we also changed the keypoints patches from 50x50 to 80x80 around each landmark. None of this changed the training much and we realized there is a problem with the model.

We also changed the number of keypoints, used a pre-trained VGG network and even changed the optimizer. We then changed the loss to be a softmax and negative log likelihood (NLL). Instead of using the blouses to test this part we used two types of classes. We used whole images with classes (like blouses, cardigans, denim, etc) instead of blouses. This trained the model well and we get the confusion matrix for the regular classes as shown in figure 4.6.

Since we see a lot of overlap between the upper body clothes like jackets with sweaters etc, we decided to test it on whole images with high-level classes (like upper, lower and full-length clothes). The confusion matrix is for high-level classes is shown in figure 4.7 and the training epoch loss graph is shown in figure 4.8. We can observe that the decrease in loss is very rapid in the initial epochs (which is expected) which we didn't see when we were using contrastive loss.



	Graphic_Tees	Denim	Pants	Sweaters	Blouses_Shirts	Leggings	Jackets_Coats	Tees_Tanks	Sweatshirts_Hoodies	Skirts	Shorts	Dresses	Cardigans	Rompers_Jumpsuits
Graphic_Tees	0.37	0.02	0.01	0.09	0.11	0.02	0.04	0.09	0.06	0.01	0.01	0.05	0.04	0.08
Denim	0.00	0.49	0.18	0.02	0.00	0.21	0.02	0.00	0.00	0.03	0.06	0.00	0.00	0.00
Pants	0.04	0.10	0.55	0.01	0.01	0.07	0.01	0.00	0.03	0.08	0.05	0.02	0.01	0.02
Sweaters	0.07	0.00	0.00	0.25	0.11	0.00	0.09	0.06	0.10	0.04	0.02	0.06	0.13	0.07
Blouses_Shirts	0.12	0.00	0.02	0.14	0.19	0.00	0.08	0.10	0.06	0.03	0.03	0.07	0.06	0.09
Leggings	0.00	0.21	0.21	0.00	0.05	0.32	0.00	0.02	0.00	0.06	0.09	0.00	0.00	0.04
Jackets_Coats	0.06	0.01	0.04	0.19	0.10	0.00	0.24	0.02	0.08	0.02	0.02	0.07	0.11	0.05
Tees_Tanks	0.14	0.00	0.01	0.11	0.13	0.00	0.07	0.20	0.04	0.02	0.05	0.07	0.05	0.08
Sweatshirts_Hoodies	0.11	0.00	0.01	0.14	0.12	0.00	0.08	0.09	0.20	0.01	0.03	0.08	0.11	0.03
Skirts	0.03	0.03	0.14	0.02	0.03	0.01	0.03	0.03	0.01	0.51	0.11	0.03	0.02	0.01
Shorts	0.03	0.04	0.09	0.02	0.03	0.02	0.01	0.02	0.01	0.12	0.54	0.01	0.02	0.03
Dresses	0.09	0.00	0.03	0.09	0.14	0.00	0.09	0.12	0.03	0.03	0.03	0.17	0.06	0.13
Cardigans	0.06	0.01	0.02	0.18	0.09	0.00	0.06	0.09	0.09	0.04	0.02	0.08	0.19	0.07
Rompers_Jumpsuits	0.12	0.00	0.00	0.06	0.09	0.01	0.04	0.07	0.05	0.04	0.04	0.14	0.05	0.29

Figure 4.6: Softmax loss with whole images on all classes: Confusion Matrix. Element in row i and column j represents the %of samples in i that was predicted as j. Darker the color more the percentages. The (i,i) are the correct predictions and the others are incorrect labelling

We then trained the softmax on blouses classes and found that the model overfits that dataset as well during training. So, essentially the Constrastive loss was the component that wasn't working.

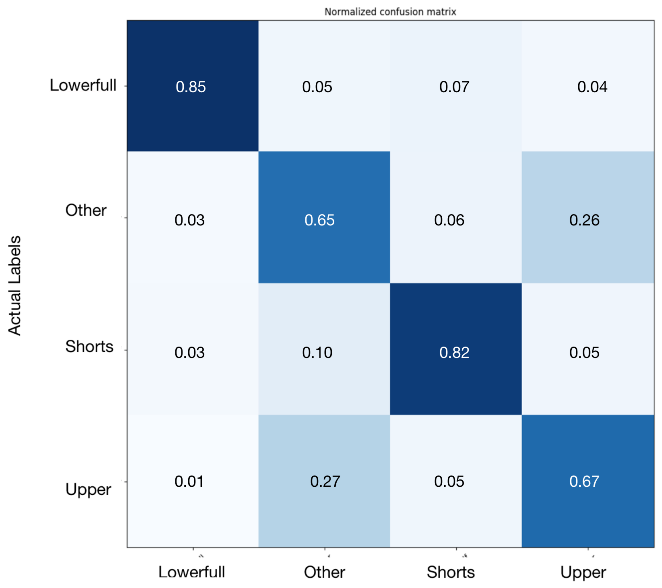


Figure 4.7: Softmax loss with whole images on high level classes classes: Confusion Matrix

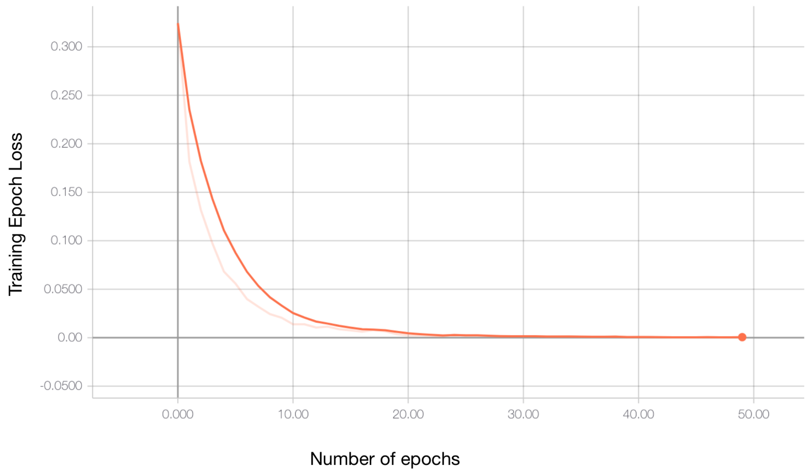


Figure 4.8: Softmax loss with whole images on high level classes classes: Confusion Matrix

We then changed the loss to the Cross entropy element-wise product loss (as defined above) and test it out on the high-classes and found that it works well for high-level images. The confusion matrix is for high-level classes with the new loss is shown in figure 4.9.

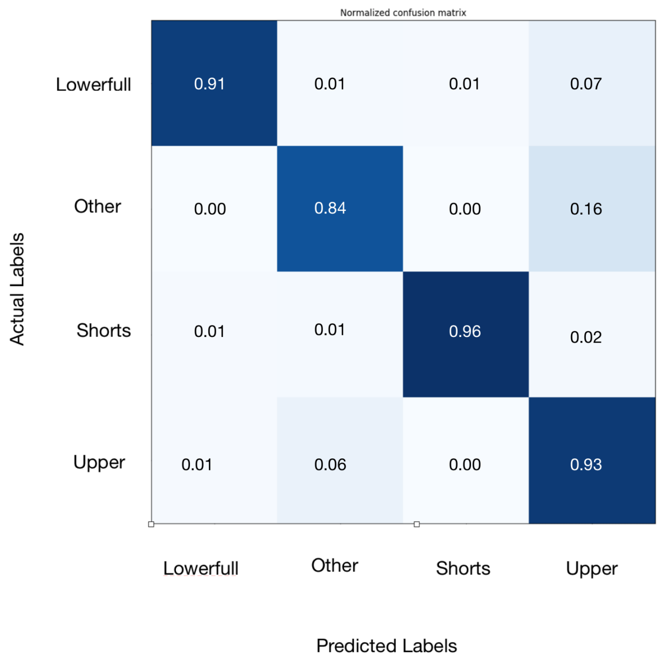


Figure 4.9: Cross-entropy element-wise product loss with whole images on high level classes classes: Confusion Matrix. The darker the (i,i) boxes the better the accuracy

We then trained the whole images with the blouses classes using the Cross-Entropy loss and got good results like 71.6% top 10 accuracy (in-depth results in the results section)

So, we went back to training the blouses with keypoints patches (both as channels and stacked over) with the Cross-Entropy loss but that didn't train well. This is when we realized we need better keypoint descriptors and we decided to use Zoom-out and hypercolumns.

The way we used these were we extracted the features from all the relu outputs for particular keypoint. We took the mean or max in case it was a patch and then concatenated across keypoints to get a single vector which was then passed to the 3 Fully connected layers to give us the d length vector. The graph of the epoch loss is shown in figure 4.10. Note that compared with figure 4.8 where we use the softmax loss, the decrease in loss is slower (which is highly likely since our loss also has a Fully connected layer which is trained as well).

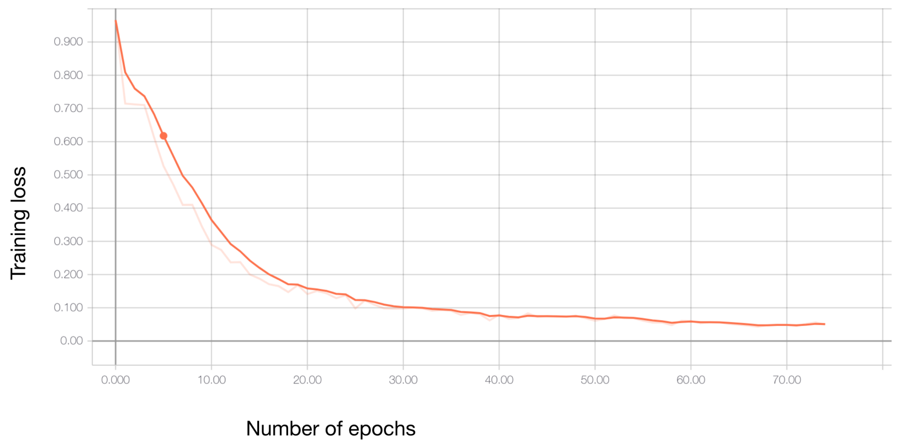


Figure 4.10: Cross-entropy element-wise product loss on blouses with keypoints: Epoch training loss

We then played around with the model with max v/s mean, Hypercolumn v/s zoomout, training the CNN network and keeping it fixed and found out that the best model was CNN with mean using Zoom-out. The results were better than the whole images and we got 79.76% top 10 accuracy (more in-depth results in the later section) which is approx 9% higher than whole images.

Once we trained this model with keypoint patch feature extraction, we also trained with the Triplet loss with three images at a time as input. We also trained the FCN network (Encoder Decoder) with Cross-entropy element-wise product loss as that network has more capability to learn compared to the VGG16 network.

Now we will compare the different feature extraction methods, loss functions we used, different CNN models, etc. We will also show the retrieval results from our best models and compare retrieval results from the whole image method and key points based method.

Figure 4.11 shows some of the top 3 retrieval results from the Zoomout VGG 16 model with mean. Figure 4.12 shows some of the failure cases of the model. From inspection of the top 3 retrieval results, we can clearly see that even though the results aren't the same clothing design, most of them match some design details or the other.

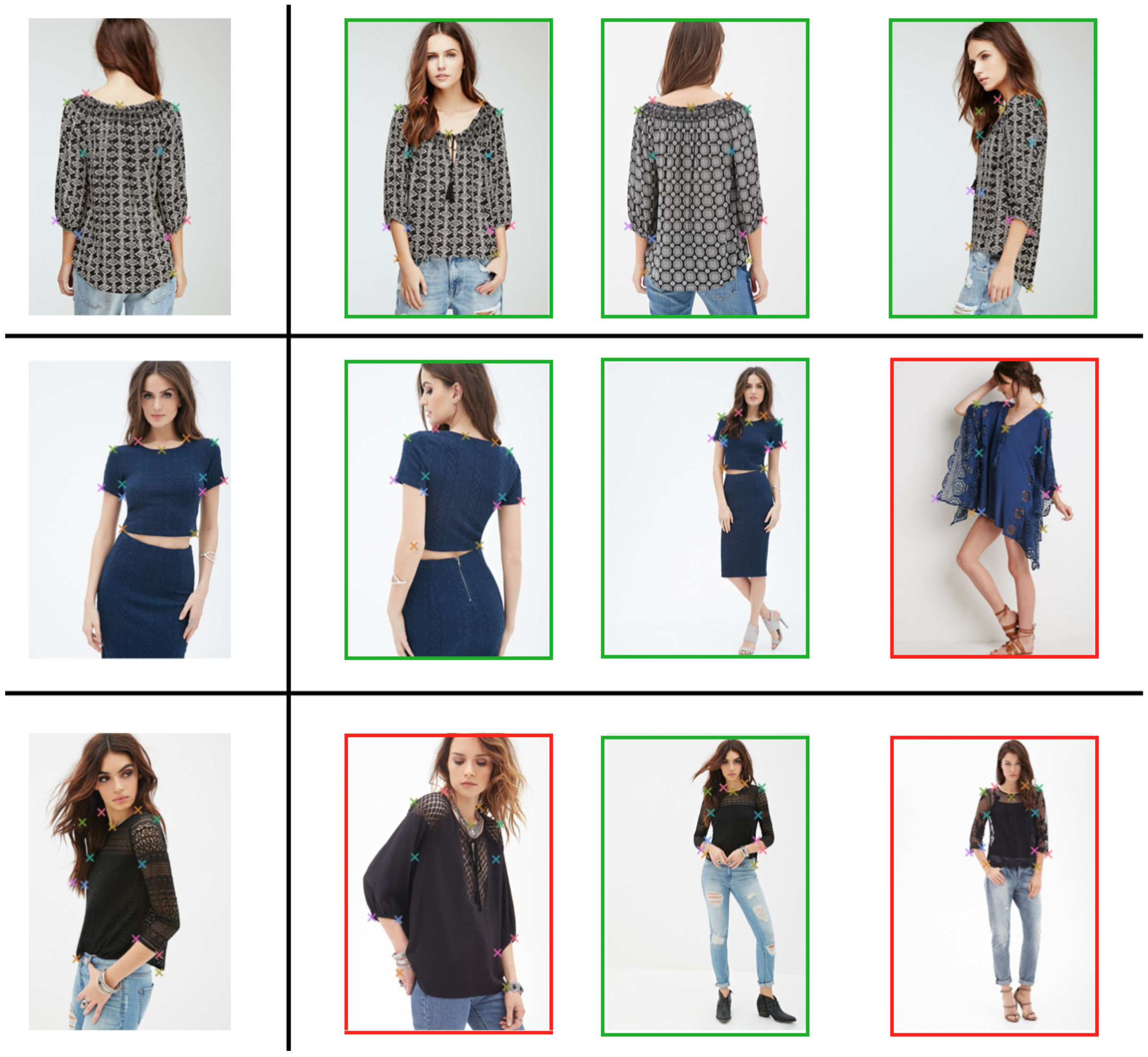


Figure 4.11: Top 3 retrieved images for keypoints with Zoom-out and mean with VGG16 and CEEP loss

Firstly in table 4.1 we compare the whole image method v/s the keypoint based method. The first row is the Whole image with VGG16 followed by 3 Fully connected layers and it was reduced to 128 length vector. The loss was common for all these models which were the Cross-Entropy Elementwise Product (CEEP) loss. The mean and max are the two way we combined the zoomout features. We can clearly see that the model with keypoints feature extraction, when allowed to train, outperforms the whole image model keeping everything else fixed. There is a 8% increase in top 3, top 5 and top 10 accuracy compared to the whole image method. We then concatenated the 128 length features from both the models and that seems to outperforms the basic whole image model by 15% in top 1 and top 3, 14% in top 5 and 11% top 10 retrieval accuracy.

Method	Top 1	Top 3	Top 5	Top 10	Top 50	Top 100
Whole image	32.44	51.99	60.83	71.66	90.16	94.27
CNN fixed with Mean	5.2	13.07	18.86	29.20	59.58	73.53
CNN trained with Max	35.36	52.42	61.20	71.54	88.41	93.64
CNN trained with Mean	38.6	60.14	69.55	79.76	93.46	96.51
CNN trained with Mean + Whole image	**47.63**	**67.87**	**74.9**	**83.0**	**94.20**	**96.88**

Table 4.1: Whole image v/s Zoomout keypoint extraction. Trained with CEEP loss. Mean refers to how patch features were combined

In table 4.2, we compare the triplet loss function with the cross-entropy element-wise product loss with the VGG16 and Zoom out keypoint extraction with mean. The Triplet loss isn't performing well at all compared to the CEEP loss.

Method	Top 1	Top 3	Top 5	Top 10	Top 50	Top 100
Triplet loss	2.73	5.04	6.78	10.46	29.01	41.03
CEEP loss	**38.6**	**60.14**	**69.55**	**79.76**	**93.46**	**96.51**

Table 4.2: Comparing different loss

In table 4.3, we compare Hypercolumn with Zoomout (mean) and see that zoom-out performs better than hypercolumn. We suspect that this is because the patch extraction can relay more information about the design than the single pixel.

Method	Top 1	Top 3	Top 5	Top 10	Top 50	Top 100
Hypercolumn	13.63	28.2	36.48	48.06	76.15	85.11
Zoomout	**38.6**	**60.14**	**69.55**	**79.76**	**93.46**	**96.51**

Table 4.3: Zoomout v/s Hypercolumn

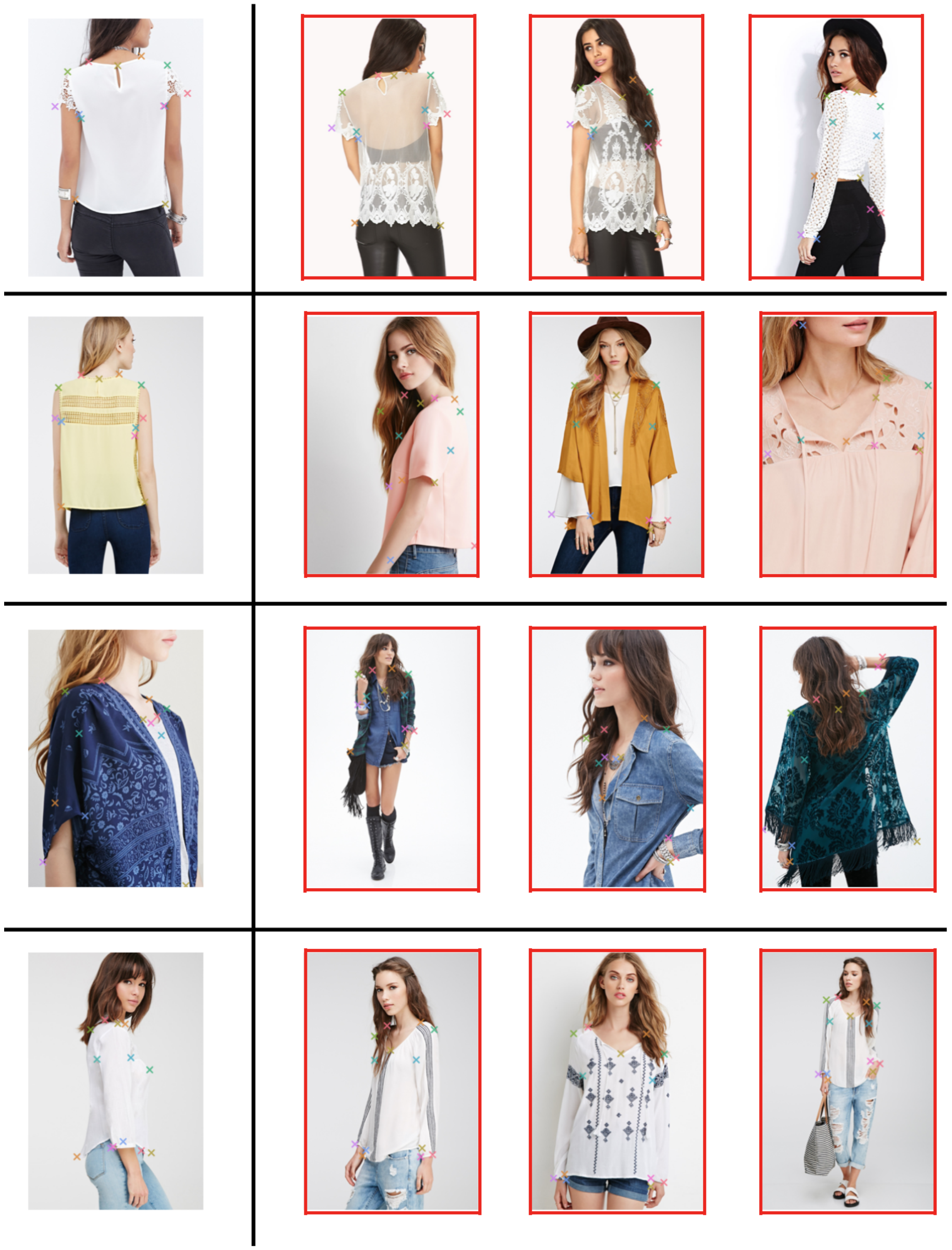


Figure 4.12: Failure cases: Top 3 retrieved images for keypoints with Zoom-out and mean with VGG16 and CEEP loss. Note that in almost all the retrieval results, we have similar design retrieved if not the same. For example, image 1's shoulder pattern, image 2's bead pattern, image 3's floral pattern

Next, we compare the retrieval results between the keypoint bases method v/s the whole image based method. In figure 4.13 we show the queries where the keypoint based method fetches the correct result but the whole image fails to.

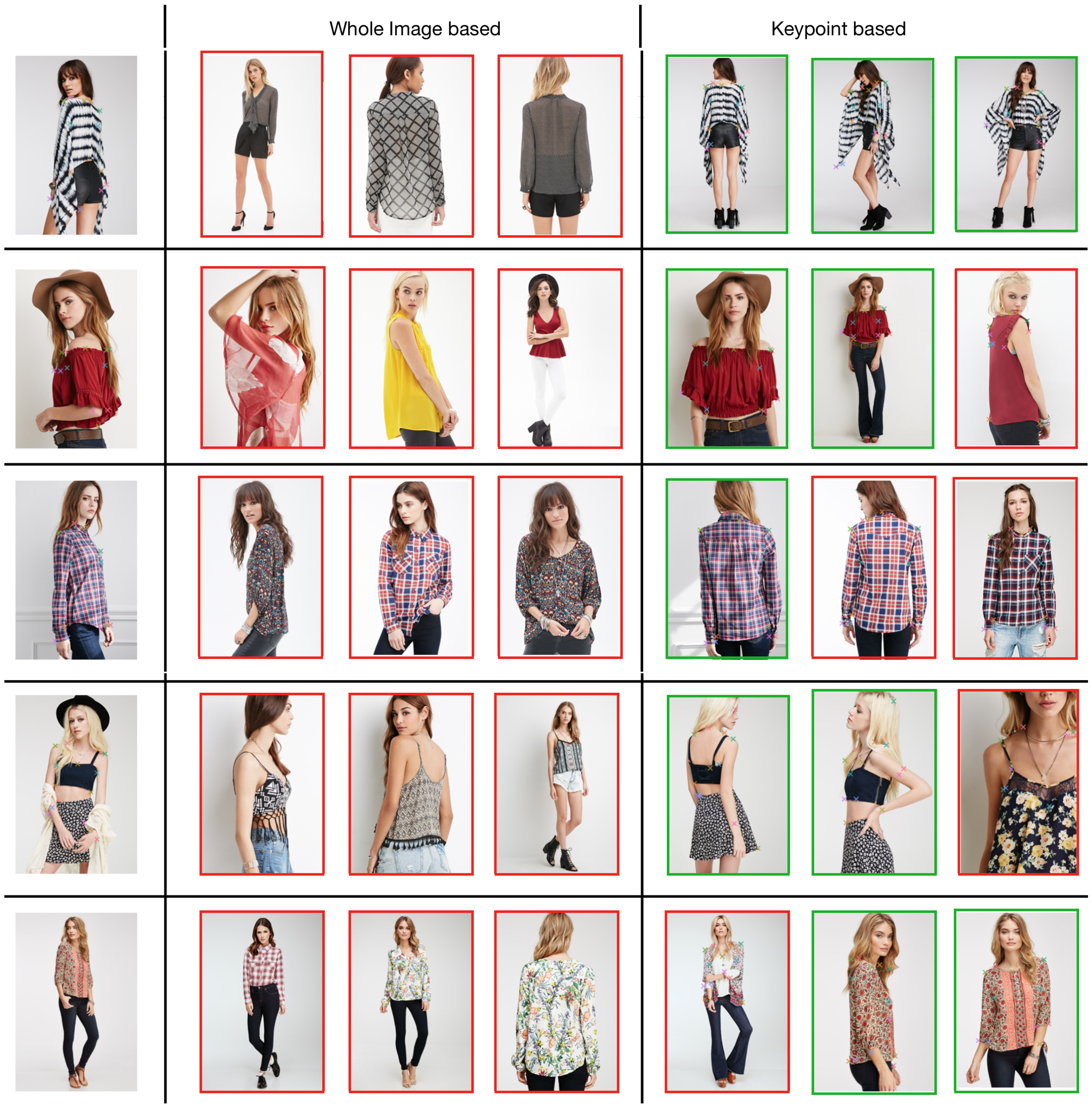


Figure 4.13: Whole image v/s keypoints with Zoom-out and mean (with VGG16 and CEEP loss). Results where kyepoints got the correct design and whole images didn't.

In table 4.4, we compare VGG16 and FCN networks. We observe that the FCN outperforms the VGG in all the top-k results except top 100. This does show that more information from the model can lead to better accuracy. Although the increase isn't as significant as we thought it might turn out to be. We also observe that for the concatenated feature vector with the whole image, the VGG performs better than FCN and that is also the best accuracy we achieved compared to all the other models we had.

Method	Top 1	Top 3	Top 5	Top 10	Top 50	Top 100
FCN	42.77	62.76	71.35	80.69	93.52	95.82
FCN + Whole image	46.26	62.7	70.29	78.51	92.34	94.76
VGG16	38.60	60.14	69.55	79.76	93.46	96.51
VGG16 + Whole image	**47.63**	**67.87**	**74.9**	**83.0**	**94.20**	**96.88**

Table 4.4: VGG16 v/s FCN

In figure 4.14, we show examples of queries where the keypoints fetched very similar designs compared to the design fetched by whole image-based features. These range from low-level designs like checks (shown in figure 4.13), beads, to higher level design like sleeve length, sleeve type (like folded v/s not folded)

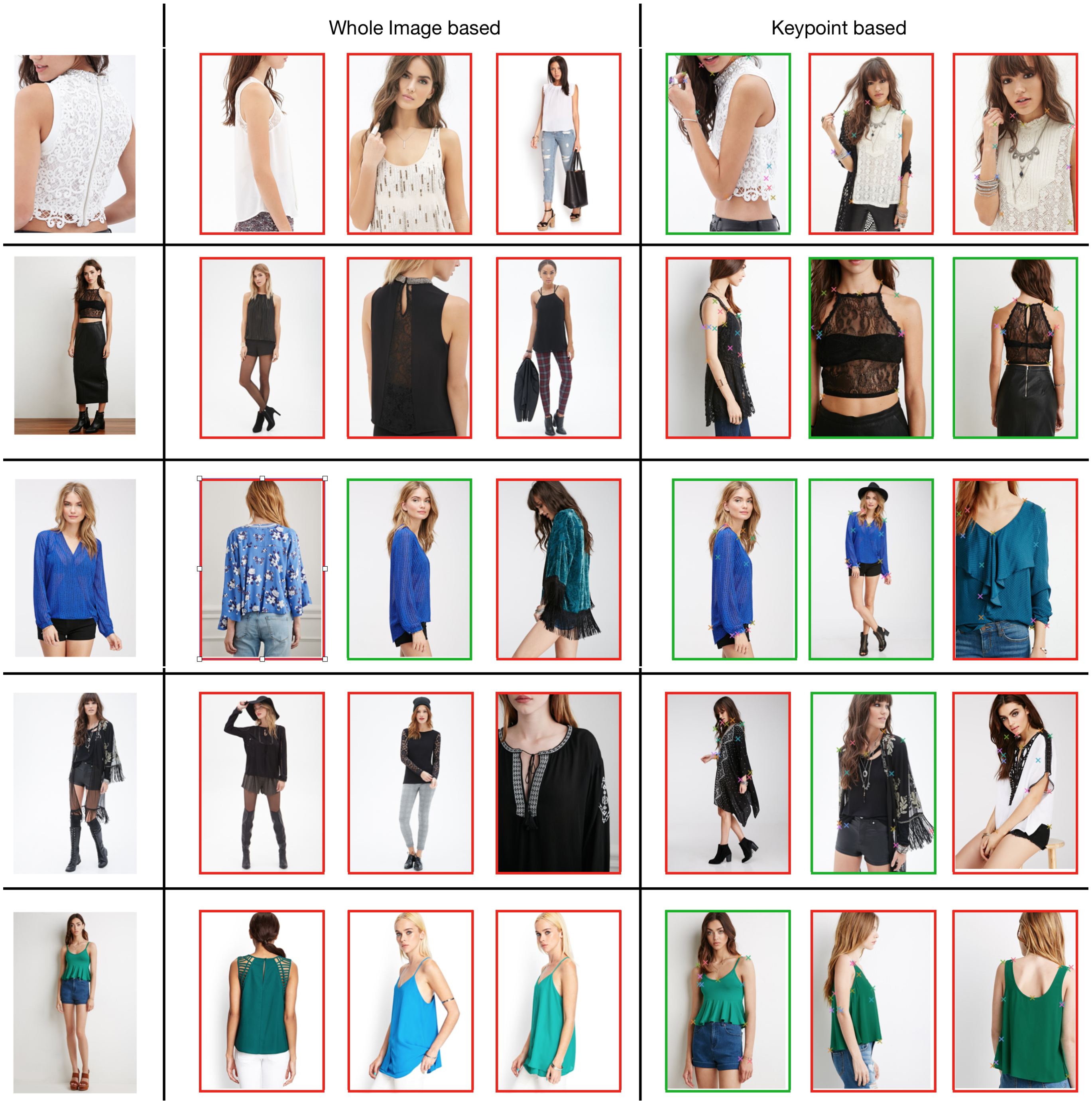


Figure 4.14: Whole image v/s keypoints with Zoom-out and mean (with VGG16 and CEEP loss). Results where keypoints get better design retrieval than whole image. For example, in the 1st, 3rd and 4th one the design is captured in all the retrieved results in keypoints. In 2nd and 5th the the high level design of the cloth like the see through top and sleeveless part is captured

CHAPTER 5: CONCLUSION

In this work, we proposed a keypoint based image retrieval method. Our model uses both high-level data and keypoint based features to focus on a more in-depth image representation which is important for clothing retrieval where the design of the cloth is one of the most crucial criteria. We also compare various ways of extracting features like Zoomout and Hypercolumn and compare a VGG network with an Encoder Decoder based CNN network (FCN). We demonstrate that the keypoint based method achieves significantly better top-k retrieval accuracy and also retrieves clothes that match in the clothing design and other aspects of the clothing like sleeve size, sleeve type, etc.

Next steps include combining the whole upper body clothing and doing retrieval with a lot more clothes. Since we couldn't find any paper that used the inshop clothing dataset to do retrieval from the DeepFashion dataset, we didn't compare our models with any other papers. So, the next step would be to set up the baseline for the whole upper body classes.

Further we would want a much bigger CNN network like ResNet152 to do feature extraction to see the capabilities of the idea of keypoints feature extraction in full scope. We also see that in some cases the landmarks aren't covering all the important aspects of the clothing and hence a better annotation may help which has some keypoints solely on the design. Also, the whole pipeline can be trained end to end so that the keypoints can be allowed to adapt accordingly.

REFERENCES

[1] Z. Liu, S. Yan, P. Luo, X. Wang, and X. Tang, “Fashion landmark detection in the wild,” in *European Conference on Computer Vision*. Springer, 2016, pp. 229–245.

[2] 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*, 2016, pp. 1096–1104.

[3] 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*, 2015, pp. 3343–3351.

[4] Alibaba and H. K. P. University, “Fashionai global challenge,” 2018. [Online]. Available: https://tianchi.aliyun.com/competition/entrance/231670/information

[5] F. Yang, A. Kale, Y. Bubnov, L. Stein, Q. Wang, H. Kiapour, and R. Piramuthu, “Visual search at ebay,” in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2017, pp. 2101–2110.

[6] D. Shankar, S. Narumanchi, H. Ananya, P. Kompalli, and K. Chaudhury, “Deep learning based large scale visual recommendation and search for e-commerce,” *arXiv preprint arXiv:1703.02344*, 2017.

[7] S. Yan, Z. Liu, P. Luo, S. Qiu, X. Wang, and X. Tang, “Unconstrained fashion landmark detection via hierarchical recurrent transformer networks,” in *Proceedings of the 25th ACM international conference on Multimedia*. ACM, 2017, pp. 172–180.

[8] Y. Chen, Z. Wang, Y. Peng, Z. Zhang, G. Yu, and J. Sun, “Cascaded pyramid network for multi-person pose estimation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7103–7112.

[9] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik, “Hypercolumns for object segmentation and fine-grained localization,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 447–456.

[10] M. Mostajabi, P. Yadollahpour, and G. Shakhnarovich, “Feedforward semantic segmentation with zoom-out features,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3376–3385.

[11] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[12] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.

[13] F. Schroff, D. Kalenichenko, and J. Philbin, “Facenet: A unified embedding for face recognition and clustering,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815–823.

[14] R. Hadsell, S. Chopra, and Y. LeCun, “Dimensionality reduction by learning an invariant mapping,” in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’06)*, vol. 2. IEEE, 2006, pp. 1735–1742.

[15] J. McAuley, C. Targett, Q. Shi, and A. Van Den Hengel, “Image-based recommendations on styles and substitutes,” in *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval.* ACM, 2015, pp. 43–52.

[16] Y. Li, L. Cao, J. Zhu, and J. Luo, “Mining fashion outfit composition using an end-to-end deep learning approach on set data,” *IEEE Transactions on Multimedia*, vol. 19, no. 8, pp. 1946–1955, 2017.

[17] H. Chen, A. Gallagher, and B. Girod, “Describing clothing by semantic attributes,” in *European conference on computer vision.* Springer, 2012, pp. 609–623.

[18] 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.* Springer, 2014, pp. 472–488.

[19] K. Yamaguchi, M. H. Kiapour, L. E. Ortiz, and T. L. Berg, “Parsing clothing in fashion photographs,” in *2012 IEEE Conference on Computer Vision and Pattern Recognition.* IEEE, 2012, pp. 3570–3577.

[20] K. Yamaguchi, M. Hadi Kiapour, and T. L. Berg, “Paper doll parsing: Retrieving similar styles to parse clothing items,” in *Proceedings of the IEEE international conference on computer vision*, 2013, pp. 3519–3526.

[21] A. C. Gallagher and T. Chen, “Clothing cosegmentation for recognizing people,” in *2008 IEEE Conference on Computer Vision and Pattern Recognition.* IEEE, 2008, pp. 1–8.

[22] Z. Hu, H. Yan, and X. Lin, “Clothing segmentation using foreground and background estimation based on the constrained delaunay triangulation,” *Pattern Recognition*, vol. 41, no. 5, pp. 1581–1592, 2008.

[23] B. Hasan and D. C. Hogg, “Segmentation using deformable spatial priors with application to clothing.” in *BMVC*, 2010, pp. 1–11.

[24] N. Wang and H. Ai, “Who blocks who: Simultaneous clothing segmentation for grouping images,” in *2011 International Conference on Computer Vision.* IEEE, 2011, pp. 1535–1542.

[25] X. Liang, C. Xu, X. Shen, J. Yang, S. Liu, J. Tang, L. Lin, and S. Yan, "Human parsing with contextualized convolutional neural network," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1386–1394.

[26] P. Sermanet, K. Kavukcuoglu, S. Chintala, and Y. LeCun, "Pedestrian detection with unsupervised multi-stage feature learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 3626–3633.

[27] M. Yang and K. Yu, "Real-time clothing recognition in surveillance videos," in *2011 18th IEEE International Conference on Image Processing.* IEEE, 2011, pp. 2937–2940.

[28] N. Inoue, E. Simo-Serra, T. Yamasaki, and H. Ishikawa, "Multi-label fashion image classification with minimal human supervision," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2261–2267.

[29] E. Simo-Serra, S. Fidler, F. Moreno-Noguer, and R. Urtasun, "Neuroaesthetics in fashion: Modeling the perception of fashionability," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 869–877.

[30] K. Hara, V. Jagadeesh, and R. Piramuthu, "Fashion apparel detection: the role of deep convolutional neural network and pose-dependent priors," in *2016 IEEE Winter Conference on Applications of Computer Vision (WACV).* IEEE, 2016, pp. 1–9.

[31] C. Corbiere, H. Ben-Younes, A. Ramé, and C. Ollion, "Leveraging weakly annotated data for fashion image retrieval and label prediction," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2268–2274.

[32] W. Di, C. Wah, A. Bhardwaj, R. Piramuthu, and N. Sundaresan, "Style finder: Fine-grained clothing style detection and retrieval," in *Proceedings of the IEEE Conference on computer vision and pattern recognition workshops*, 2013, pp. 8–13.

[33] J. Huang, R. S. Feris, Q. Chen, and S. Yan, "Cross-domain image retrieval with a dual attribute-aware ranking network," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1062–1070.

[34] W.-L. Hsiao and K. Grauman, "Learning the latent look: Unsupervised discovery of a style-coherent embedding from fashion images," in *2017 IEEE International Conference on Computer Vision (ICCV).* IEEE, 2017, pp. 4213–4222.

[35] X. Han, Z. Wu, Y.-G. Jiang, and L. S. Davis, "Learning fashion compatibility with bidirectional lstms," in *Proceedings of the 25th ACM international conference on Multimedia.* ACM, 2017, pp. 1078–1086.

[36] T. Nakamura and R. Goto, "Outfit generation and style extraction via bidirectional lstm and autoencoder," *arXiv preprint arXiv:1807.03133*, 2018.

[37] E. Simo-Serra and H. Ishikawa, "Fashion style in 128 floats: Joint ranking and classification using weak data for feature extraction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 298–307.

[38] M. I. Vasileva, B. A. Plummer, K. Dusad, S. Rajpal, R. Kumar, and D. Forsyth, "Learning type-aware embeddings for fashion compatibility," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 390–405.

[39] Z. Al-Halah, R. Stiefelhagen, and K. Grauman, "Fashion forward: Forecasting visual style in fashion," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 388–397.

[40] S. Zhu, R. Urtasun, S. Fidler, D. Lin, and C. Change Loy, "Be your own prada: Fashion synthesis with structural coherence," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 1680–1688.

[41] A. Raj, P. Sangkloy, H. Chang, J. Lu, D. Ceylan, and J. Hays, "Swapnet: Garment transfer in single view images," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 666–682.

[42] D. Ramanan and X. Zhu, "Face detection, pose estimation, and landmark localization in the wild," in *2012 IEEE conference on computer vision and pattern recognition*. IEEE, 2012, pp. 2879–2886.

[43] J. J. Tompson, A. Jain, Y. LeCun, and C. Bregler, "Joint training of a convolutional network and a graphical model for human pose estimation," in *Advances in neural information processing systems*, 2014, pp. 1799–1807.

[44] W. Wang, Y. Xu, J. Shen, and S.-C. Zhu, "Attentive fashion grammar network for fashion landmark detection and clothing category classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4271–4280.

[45] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[46] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *international Conference on computer vision & Pattern Recognition (CVPR'05)*, vol. 1. IEEE Computer Society, 2005, pp. 886–893.

[47] D. Frossard, "VGG in TensorFlow," https://www.cs.toronto.edu/~frossard/post/vgg16/, 2016, [Online].

[48] Y. Nanda, "What is the VGG neural network?" https://qr.ae/TW8pFU, 2018, [Online].

[49] A. Katte, "Choosing Between GAN Or Encoder Decoder Architecture For ML Applications Is Like Comparing Apples To Oranges," https://www.analyticsindiamag.com/choosing-between-gan-or-encoder-decoder-architecture-for-ml-applications-is-like\-comparing-apples-to-oranges/, 2018, [Online].

[50] Wikipedia, "k-nearest neighbors algorithm," https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm, 2018, [Online].