Large Scale Fashion Search System with Deep Learning and Quantization Indexing

Thoi Hoang Dinh
Framgia Inc
Ha Noi, Viet Nam
hoang.dinh.thoi@framgia.com

Toan Pham Van
Framgia Inc
Ha Noi, Viet Nam
pham.van.toan@gmail.com

Ta Minh Thanh
Le Quy Don Technical University
Ha Noi, Viet Nam
thanhtm@mta.edu.vn

Hau Nguyen Thanh
Framgia Inc
Ha Noi, Viet Nam
nguyen.thanh.hau@framgia.com

Anh Pham Hoang
Framgia Inc
Ha Noi, Viet Nam
pham.hoang.anh@framgia.com

ABSTRACT

Recently, the problems of clothes recognition and clothing item retrieval have attracted a number of researchers, due to its practical and potential values to real-world applications. The main task is to automatically find relevant clothing items given a single user-provided image without any extra metadata. Most existing systems mainly focus on clothes classification, attribute prediction, and matching the exact in-shop items with the query image. However, these systems do not mention the problem of latency period or the amount of time that users have to wait when they query an image until the query results are retrieved. In this paper, we propose a fashion search system that automatically recognizes clothes and suggests multiple similar clothing items with an impressively low latency. Through extensive experiments, it is verified that our system outperforms almost existing systems in term of clothing item retrieval time.

CCS CONCEPTS

• **Information systems** → **Search engine architectures and scalability**; *Search engine indexing*;

KEYWORDS

Fashion Search System, Clothes Recognition, Image Similarity Learning, Quantization Indexing

ACM Reference Format:
Thoi Hoang Dinh, Toan Pham Van, Ta Minh Thanh, Hau Nguyen Thanh, and Anh Pham Hoang. 2018. Large Scale Fashion Search System with Deep Learning and Quantization Indexing. In *The Ninth International Symposium on Information and Communication Technology (SoICT 2018), December 6–7, 2018, Danang City, Viet Nam*. ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3287921.3287964

SoICT 2018, December 6–7, 2018, Danang City, Viet Nam
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-6539-0/18/12...$15.00
https://doi.org/10.1145/3287921.3287964

1 INTRODUCTION

Nowadays, online shopping has become much more popular than in-store shopping because of its convenience. The shoppers can browse a shopping website to buy anything without leaving their comfortable houses. It makes the online shopping a very potential market. According to the Shopifyplus[1], the online shopping worldwide revenue is expected to rise from $481.2 billion in 2018 to $712.9 billion by 2022 and much of the revenue comes from clothing item shopping.

Despite the huge convenience of online shopping, online shoppers usually encounter the problem of information overload. Shoppers often difficultly find the desired clothing items to their needs, since there are many online shopping websites. In literature, the problem of in-shop clothes retrieval has been recently mentioned in [15]. The main challenge lies in the fact that photos of clothing items taken by users in real-world situations are often different from photos on online shopping websites. Photos from online shopping websites are professionally taken under controlled settings, whereas photos from users are often taken under uncontrolled settings and at lower resolution. Moreover, there are large variations of clothing items in style, texture, and cutting. Therefore, the problem of in-shop clothes retrieval is a real challenge to researchers. A number of systems have been built to address this problem. Though these methods achieve good accuracy, the searching for similar images in large database and the retrieval time of the system are hardly mentioned.

In this paper, we focus on the specific issue of existing fashion search systems. It is the latency time when searching in large image database. When users want to search a similar clothing items, they often expect the query results to be retrieved soon. Therefore, we propose a new fashion search system based on state-of-the-art method to address this problem. The workflow of our system is described as in Fig. 1. There are four stages including: object detection on mobile device, generating image embedding, quantization indexing, and searching for similar images in a large database.

[1] https://www.shopify.com/enterprise/ecommerce-fashion-industry

In this paper, two new features are introduced to make our system distinct from other systems. (1) The clothes recognition network is trained on a high-end computer as usual, and the trained model is, then, deployed on mobile device itself. (2) A new indexing method is designed to effectively represent image embedding in a new space. Further details of the indexing method will be later explained in 4.3. Finally, the searching for similar images in large image database is carried out with the well-known Elasticsearch engine [5].

In summary, the main contribution of this paper is the proposal of a fashion search system with two new added features and state-of-the-art methods. Extensive experiments are carried out to support the superiority of our system in term of real-time retrieval. The proposed system are well capable of suggesting multiple clothing items based on a query image at a short delay. As such, our system can meet the requirements of real-time applications in real-world situations.

The rest of this paper is organized as follows. Section 2 presents a brief review of related works. The procedure of system setup for experiment and data collecting and processing for both training and evaluating phase is mentioned in Section 3. In Section 4, our proposed method is thoroughly discussed. Experimental results and evaluation are presented in Section 5 to support the superiority of our system to all other systems in term of real-time retrieval. Finally, we conclude the paper in Section 6.

2 RELATED WORKS

2.1 Clothes recognition

The problem of clothes recognition has attracted tremendous attention from researchers for over the past decade. Many approaches have been proposed to address this problem. Earlier works mainly use heavily hand-crafted features from images such as SIFT, HOG, and color information. These features are, then, employed to train different machine learning algorithms. For instance, in [2], SURF, HOG, and color information were used as learning features; or in [3], the clothes recognition system used four types of features including SIFT, texture descriptors from the Maximum Response Filters, color in the LAB space, and skin probabilities. In term of machine learning algorithms, these systems above used simple methods as classifier. [2] employed Random Forests algorithm for clothes type classification and linear SVM algorithm as visual attribute classifier, [3] also used SVM algorithm for learning clothing attributes. However, the systems based on hand-crafted features have some drawbacks due to the limited power of these features. Recently, a number of systems [8] [12] [17] have adopted end-to-end deep learning technique in order to learn more distinct representation of clothes image and mitigate the variations of cross-scenarios and pose types. Some new network architectures such as Dual Attribute-aware Ranking Network (DARN) [12] and FashionNet [17] have been proposed to address both clothes recognition and clothing item retrieval problem with a single deep network. It's noteworthy that all clothes recognition systems of previous related works were built to run on desktop. However, in our approach, the neural network for clothes recognition is trained with a high-end computer and, the trained model is exported to mobile-ready version to run on mobile device. It enables the whole system to achieve low-latency retrieval time.

2.2 Clothing retrieval

The task in clothing retrieval is to find relevant clothing items given a user-provided image automatically. Some related works tackled the clothing retrieval problem using global or fine-grained attribute prediction [26], or using parsing [27]. Recently, deep convolutional neural network has been successfully applied to feature representation. In literature, many deep models have been built for image similarity learning. Most of them are trained with the tripletloss function [25]. In general, each image is represented by an unique embedding vector by a deep convolutional neural network. During the training phase, the parameters of the network are fine-tuned, so that Euclidean distance between embeddings of similar images are smaller and much larger for embeddings of different images. For instance, M. Hadi Kiapour *et al.* applied the ImageNet pre-trained model [14] as the base network and two fully-connected layers (4096 dimensions) for feature representation [8]. Since [8] directly used the pre-trained ImageNet features, their method is not suitable for clothing items representation. In [17], Ziwei Liu *et al.* defined a new network structure named FashionNet, which is similar to VGG-16 [22]. The last convolutional layer in FashionNet consisted three components and was designed to capture both global and local features of the image, as well as, to predict landmarks' locations and their visibility. Though the accuracy of clothes retrieval is high in [17], the FashionNet is not suitable for real-time clothes retrieval system because there are too many components in a single network. In our system, the network for clothes retrieval is based on the Inception network [23]; for feature representation, we use only 128 dimensions instead of 4096 dimensions as in previous works. That fact enables our system to achieve impressively low latency and meet the requirements of real-time retrieval.

2.3 Indexing methods

Similarity search or nearest neighbor search in large database is a core task in many applications such as data analytics, data processing, and multimedia content analyzing. The task is to find an item from a large database that is nearest to the query item. This task can be defined as follows: given a set of items $X = \{x_1, x_2, ..., x_N\}$ and a query item q, the job is to find $n \in X$, which is the nearest neighbor of q so that the distance between n and q is minimum across all items in the dataset. Euclidean distance is often employed as the distance metric, however, l_1 distance and cosine similarity are also possible.

The nearest neighbor search in large database is extremely challenging because computing the distance between the query item and each item in database in high dimensional space can be costly. Moreover, nearest neighbor search is often

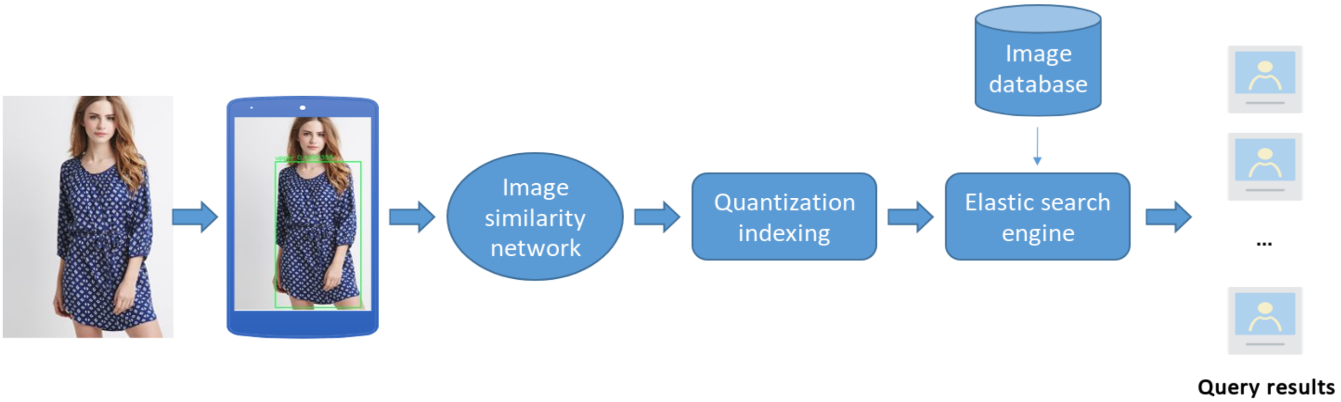

Figure 1: The architecture of the fashion search system.

time-consuming, especially when the size of the database is large. In literature, this problem has been well explored [24]. Instead of finding the exact nearest neighbor, a number of methods have been proposed to approximately find the nearest neighbor. Most of these methods try to transform data from the original space into a lower space where the nearest neighbor search can be accurately carried out. Their computational cost and time are greatly reduced.

3 IMAGE DATASET AND SYSTEM SETUP

3.1 Image dataset for training

Since data is the utmost important factor in any machine learning tasks, the data preparation procedure must be carried out carefully. As for training the clothes detection network, we specifically use the well-known dataset from DeepFashion [17] and the street-to-shop dataset from [8]. The main reason to choose these datasets is that they are from different domains and have a large number of images with bounding box annotations. All images from both DeepFashion and street-to-shop are annotated with bounding box information and type of clothes. In this experiment, we classify clothes into three categories, namely upper clothes, lower clothes, and full-body clothes. As such, each image is labeled with one category and four parameters, which define the bounding box: $x_{min}, x_{max}, y_{min}, y_{max}$. After merging all images from the two datasets above, we end up with the final dataset of about $445K$ images in total. 90% of the final dataset goes for training, and 10% goes for validation.

As for training the image similarity network, we also use data from the street-to-shop dataset. All images are well categorized and stored in different folders so that images from the same directory are of similar clothing items and images from different directories are of different clothing items. There are $175K$ images in total, and about $545K$ triplets are generated for training.

3.2 Image dataset for evaluation

In the evaluation phase, we specifically use data of the In-Shop Clothes Retrieval benchmarks from DeepFashion dataset [17] to evaluate both the retrieval time and the retrieval accuracy with the top-k recall rate as in [7]. The main reason to choose this dataset is that it contains a relatively large number of images of clothing items for both men and women. This dataset also contains clothing items from different categories such as shirts, cardigans, leggings, pants, tees tanks and, sweaters. There are 52712 images in total including 8081 query images and 44631 corresponding relevant images (about 5.5 relevant items per query image). As such, this dataset can be effectively used to evaluate our proposed indexing method and compare it with other indexing methods.

In order to support the superiority of the proposed indexing method in large scale application, we use two additional datasets including the Oxford building dataset [18] and the MIRFLICKR retrieval dataset [13]. The Oxford building dataset consists of 5062 images collected from Flickr[2] by searching for particular Oxford landmarks. The collection has been manually annotated to generate a comprehensive ground truth for 11 different landmarks, each represented by 5 possible queries. This gives a set of 55 queries over which an object retrieval system can be evaluated. The MIRFLICKR retrieval dataset is offered by the LIACS Medialab at Leiden University, The Netherlands. This is the largest image dataset in our experiment, which contains up to one million images. For each of the two additional datasets, we also analyze the retrieval accuracy with the top-k recall rate and the retrieval time among different indexing methods.

3.3 System setup

Our experiment is conducted on a computer with Intel Core i5-7500 CPU @3.40GHz 4 cores, 32GB of RAM, GPU GeForce GTX 1080 Ti, and 1TB SSD Harddisk. We use Elasticsearch version 2.3.1 which runs on Java 9. The operating system

[2] https://www.flickr.com/

is Ubuntu 16.04 64 bit. Both the clothes recognition network and image similarity network are implemented with the well-known TensorFlow framework [1]. The source code for Locality Sensitive Hashing (LSH) scheme is from this repository[3] on Github.

4 OUR APPROACH

4.1 Clothes recognition on mobile device

Clothes recognition is the first item in our workflow. Its task is to identify the bounding box where clothing item is presented. This process is necessary since we are only interested in the region where clothing item is presented. In literature, the problem of object detection has been well explored. Multiple network structures have been proposed to address this problem such as Fast R-CNN [6], Faster R-CNN [20], and YOLO [19]. Though Fast R-CNN [6], Faster R-CNN [20] gain good accuracy, they cannot be applied to real-time applications since they are too slow. The YOLO [19] and Fast YOLO [21] can run at high FPS, however, their accuracy is relatively low when compared to other methods. In [16], Wei Liu *et al.* proposed a new network structure called Single Shot MultiBox Detector (SSD) that can both enhances the detection accuracy and reduce the detection time. On the Pascal VOC2007 test, SSD framework (VGG16 as the base network) achieved 74.3% mAP at 46 FPS whereas the Faster R-CNN, YOLO, and Fast YOLO can only achieved 73.2%, 66.4%, 52.7% at 7 FPS, 21 FPS, 155 FPS respectively [16]. In our experiment, the MobileNet [10] is adopted as the base network since MobileNet is a light weight convolutional neural network which is specially built for mobile and embedded applications. Employing MobileNet SSD as the clothes recognition network can result in better accuracy and performance because it utilizes the merits of both the SSD framework and the MobileNet.

In our experiment, clothing items are classified into three categories, namely upper clothes, lower clothes, and full-body clothes. The methodology to train the MobileNet SSD is the same as in [16]. After the training phase is done, the trained model is, then, exported to mobile-ready version with TensorFlow Mobile framework. Deploying clothes recognition network on mobile device can make full use of the computational resource of a mobile device and share a lot of load with the backend server. Therefore, it can help to improve the retrieval time of the system.

4.2 Image similarity network

Earlier works on image similarity usually employ the VGG-16 [22] as the base network and add one or two fully-connected layer for feature representation. It is noteworthy that in previous works, the embedding of each image is a 4096-dimensional vector. Though the mentioned method results in good accuracy in term of clothes retrieval, it is not suitable for real-time application since the embedding vector is too large. In our approach, we use the Inception network [23] as the base network and two fully-connected layer for feature representation. The architecture of our image similarity network is described as in Fig. 2.

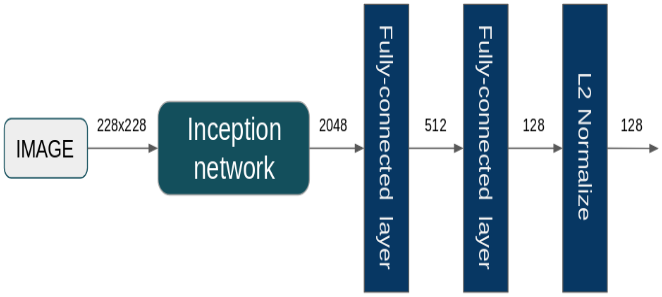

Figure 2: Image similarity network.

The Inception network has been widely and successfully used in many computer vision tasks. The reason is both the width and depth of the Inception network are increased while the computational budget is kept constant. In fact, the Inception network has outperform the VGG-16[22] in many task such as the ILSVRC 2014 Classification Challenge [23] and the ILSVRC 2014 Detection Challenge [23]. As such, we choose the Inception network over the VGG-16[22] to be the base network. As for feature representation, we use two fully-connected layers with 512 neurons for the first layer and 128 neurons for the second layer. Therefore, the embedding of an image is a 128-dimensional vector. With this approach, important features of the image are still kept whereas the retrieval speed of the system is significantly improved because the size of the embedding vector is greatly reduced.

The methodology to train our image similarity network is the same as in [25]. The goal is to train an image similarity model via a large number of triplets. A triplet is a set of three images denoted by $t_i = (p_i, p_i^+, p_i^-)$ where p_i is the anchor image, p_i^+ is the positive image, and p_i^- is the negative image. In this experiment, both anchor image and positive image are of the same or similar clothes whereas the negative image contains completely different clothing item from that of the anchor image. The similarity between two image A and B is defined according to their Euclidean distance in the embedding space:

$$D(f(A), f(B)) = \|f(A) - f(B)\|_2^2 \tag{1}$$

where $f(.)$ is the embedding function and $D(.,.)$ is the Euclidean distance in this space. The triplet loss function of t_i can, then, be defined as follows:

$$L(p_i, p_i^+, p_i^-) = max(0, m + D(f(p_i), f(p_i^+)) - D(f(p_i), f(p_i^-))) \tag{2}$$

where m is the margin parameter of the distance between (p_i, p_i^+) and the distance between (p_i, p_i^-). Specifically, the margin parameter m is chosen to be 1.0 in our experiment. During the training phase, all the parameters of the network are fine-tuned to minimize the triplet loss function $L(p_i, p_i^+, p_i^-)$. In other words, the objective is to make the

[3] https://github.com/kayzhu/LSHash

Euclidean distance between anchor image and positive image as small as possible and the distance between anchor image and negative image as large as possible.

4.3 Quantization Indexing

As mentioned above, the image similarity network takes an image as input and returns a feature representation of that image. It is noteworthy that the representation of each image is a 128-dimensional vector. The searching for similar images can be done by comparing the similarity metric (cosine similarity or Euclidean distance) between the query image and each of the images in the database. This approach is known as the brute force search approach. The main drawback of this method is that its speed is slow and will grow exponentially when the size of the image database grows. That fact, along with the increasing demands on real-time retrieval in real-world applications, requires a new indexing method to minimize the processing time while searching in large image database.

Quantization indexing is an indexing method that transforms the image embedding vector from numerical space to text space. Then, the searching for similar images in database can be carried out by a full text search engine. The proposed indexing method has to meet the following criteria:

- The properties of the feature representation of an image must remain unchanged after being indexed by the proposed method. In other word, the searching for similar images in numerical space (before being indexed) and text space (after being indexed) must return the same results.
- Words which are generated by the proposed indexing method in the text space must have a suitable length so that the trade-off between accuracy and latency is acceptable.

Based on the mentioned criteria, we propose a new indexing method called quantization indexing, which uses Q as the quantization factor to transform the embedding vector from numerical space to text space. Let the embedding vector (output of the image similarity network) be denoted by $w = [w_1, w_2, ..., w_n]$ where n is the length of the embedding vector ($n = 128$ in this case), then, the encoding function is given in the following formula:

$$encode(w) = \bigcup_{i}^{n} \bigcup_{j}^{floor(|Q*w_i|))} \sigma_i \tag{3}$$

where Q is the quantization factor, Q is equal or greater than 1 and σ_i is defined as follows:

$$\sigma_i = converter(w_i) = \begin{cases} ”p\_” + i & if \quad w_i > 0 \\ ”n\_” + i & if \quad w_i < 0 \end{cases} \tag{4}$$

For each element w_i in the feature vector w, the term σ_i is repeated for a $floor(|Q * w_i|)$ times. For instance, given $Q = 50$, the feature vector $w = [0.02, -0.04, 0.06]$, then $|Q * w_i|$ will be $[1, 2, 3]$ and encoded vector in text space is **p1 n2 n2 p3 p3 p3**. With quantization indexing, all image embedding vectors are transformed into text space and the searching for similar images can be easily and effectively carried out with Elasticsearch engine.

The quantization factor Q has a crucial impact on the accuracy and performance of this indexing method. If Q is a small value, then, the length of the indexed vector will also be small. Thus, the retrieval time will decrease and the accuracy will, also, decrease. In contrast, when the value of Q is large, the length of the indexed vector will also be large. Therefore, both the retrieval time and accuracy will increase. The value of Q is a human chosen convention and varies with different dataset. As such, the value of the quantization factor Q must be carefully chosen.

5 RESULTS AND EVALUATION

In this section, we further analyze the retrieval accuracy (mean average precision at top-k) and the retrieval time of our proposed indexing method with other indexing methods. Specifically, we compare our proposed method with the brute-force search approach and the Locality Sensitive Hashing scheme [4].

5.1 Mean average precision and retrieval time on the In-Shop Clothes Retrieval benchmarks

In this subsection, we compare our proposed indexing method with the traditional brute-force search method and the LSH scheme. Images used in this experiment are from the In-Shop Clothes Retrieval benchmarks of the DeepFashion dataset. The mean average precision is measured at top-10 retrieval accuracy. For the brute-force search, both Euclidean distance and cosine similarity are used as evaluation metric and for the LSH technique, the Euclidean distance is used. As mentioned earlier, the quantization factor Q has strong impact on both accuracy and performance of the proposed quantization indexing method. Therefore, in this experiment, we choose to use three different quantization factors $Q \in \{10, 50, 100\}$ to fully investigate the performance of our proposed method. Details on the retrieval accuracy and retrieval time are shown in Table 1.

It can be easily observed that the brute-force search method achieves the highest mAP (79.36 % and 76.42 % for cosine similarity and Euclidean distance respectively). However, the retrieval time of this method is also highest since this approach computes distance between query item and all items in database directly in the original embedding vector space. The LSH scheme has a slightly worse retrieval accuracy (72.32% mAP) than the brute-force search but the retrieval time is significantly reduced from 2.178 seconds (brute-force with cosine similarity) and 2.982 seconds (brute-force with Euclidean distance) down to 0.068 s. As for the proposed method, we can notice that both the retrieval time and retrieval accuracy increase as Q increases. This can be easily explained because when the value of Q is small, the length of the indexed vector in the text space is small and a lot of information is lost after indexing. When the value of Q is large, the length of the indexed vector in the text space is large, which leads to higher accuracy and also more time for searching. When $Q = 50$, the accuracy of the proposed method is 76.45%, which is better than the LSH scheme (72.32%) and the retrieval time is improved from 0.068 to 0.024 second ($\approx$ 64% faster). When $Q = 100$, our quantization indexing method achieves comparable accuracy when compared to brute-force search approach (78.21% and 79.36%) while the retrieval time is greatly reduced from 2.178 to 0.032 second.

In summary our proposed indexing method can significantly improve the retrieval time while the retrieval accuracy is still kept at a high rate. Moreover, our proposed method can can outperform the LSH scheme in term of both retrieval accuracy and retrieval time.

5.2 Mean average precision and retrieval time on the Oxford building dataset and the MIRFLICKR retrieval dataset

In previous experiment, we use our image similarity network for evaluating the proposed indexing method. In this experiment, we use different pre-trained models including VGG-16 [22], VGG-19 [22], ResNet50 [9], and DenseNet121[11] for generating image embedding vector. It is noteworthy that image embedding vector are of different dimensions instead of 128 dimensions as previously. Specifically, image embedding generated by VGG-16 [22] and VGG-19 [22] is a 4096-dimensional vector. Embedding by ResNet50 [9] and DenseNet121 [11] is a 2048, 1664-dimensional vector respectively. In this time, the mean average precision is measured at top-5 retrieval accuracy.

As for the Oxford building dataset, our proposed method is, again, compared with the brute-force search approach and the LSH scheme. The retrieval accuracy is much lower than that in previous experiment since all the pre-trained model are not trained with the triplet loss function. However, it is noticeable that our proposed method (with $Q = 100$)can achieve relatively high retrieval accuracy when compared to the brute-force search approach regardless of pre-trained

Table 1: Mean average precision and retrieval time on the In-Shop Clothes Retrieval benchmarks

Neural network model	Indexing methods	mAP (%)	Retrieval time (s)
Our image similarity network (128-dimensions)	Brute-force search (cosine similarity)	79.36	2.178
	Brute-force search (Euclidean distance)	76.42	2.982
	LSH (8bits)	72.32	0.068
	Quantization indexing ($Q = 10$)	68.12	0.013
	Quantization indexing ($Q = 50$)	76.45	0.024
	Quantization indexing ($Q = 100$)	78.21	0.032

Table 2: Mean average precision and retrieval time on the Oxford building dataset

Pre-trained model	Indexing methods	mAP (%)	Retrieval time (ms)
VGG16	Brute-force search (cosine similarity)	26.24	0.132
	Brute-force search (Euclidean distance)	24.23	0.129
	LSH (8bits)	18.54	0.038
	Quantization indexing ($Q = 10$)	8.4	0.012
	Quantization indexing ($Q = 50$)	21.08	0.023
	Quantization indexing ($Q = 100$)	25.04	0.048
VGG19	Brute-force search (cosine similarity)	22.06	0.122
	Brute-force search (Euclidean distance)	20.48	0.134
	LSH (8bits)	16.43	0.042
	Quantization indexing ($Q = 10$)	13.42	0.012
	Quantization indexing ($Q = 50$)	18.32	0.024
	Quantization indexing ($Q = 100$)	19.21	0.041
ResNet50	Brute-force search (cosine similarity)	26.77	0.126
	Brute-force search (Euclidean distance)	25.43	0.123
	LSH (8bits)	27.37	0.062
	Quantization indexing ($Q = 10$)	13.43	0.013
	Quantization indexing ($Q = 50$)	24.32	0.024
	Quantization indexing ($Q = 100$)	26.21	0.032
DenseNet121	Brute-force search (cosine similarity)	23.42	0.128
	Brute-force search (Euclidean distance)	22.98	0.127
	LSH (8bits)	18.32	0.048
	Quantization indexing ($Q = 10$)	11.32	0.015
	Quantization indexing ($Q = 50$)	15.43	0.032
	Quantization indexing ($Q = 100$)	19.32	0.046

Table 3: Mean average precision and retrieval time on the MIRFLICKR retrieval dataset

Pre-trained model	Indexing methods	mAP (%)	Retrieval time (ms)
VGG16	Brute-force search (cosine similarity)	49.3	208.923
	Brute-force search (Euclidean distance)	48.23	210.934
	LSH (8bits)	44.54	3.3240
	Quantization indexing ($Q = 10$)	17.54	0.2338
	Quantization indexing ($Q = 50$)	35.21	0.4342
	Quantization indexing ($Q = 100$)	46.72	0.9376
VGG19	Brute-force search (cosine similarity)	49.3	229.147
	Brute-force search (Euclidean distance)	44.74	220.321
	LSH (8bits)	42.16	3.2950
	Quantization indexing ($Q = 10$)	13.42	0.2354
	Quantization indexing ($Q = 50$)	26.85	0.3265
	Quantization indexing ($Q = 100$)	48.21	0.8623
ResNet50	Brute-force search (cosine similarity)	47.25	248.62
	Brute-force search (Euclidean distance)	45.78	234.09
	LSH (8bits)	44.43	3.408
	Quantization indexing ($Q = 10$)	18.65	0.2351
	Quantization indexing ($Q = 50$)	28.71	0.4561
	Quantization indexing ($Q = 100$)	43.42	0.8721
DenseNet121	Brute-force search (cosine similarity)	48.14	245.29
	Brute-force search (Euclidean distance)	45.91	239.07
	LSH (8bits)	42.11	3.258
	Quantization indexing ($Q = 10$)	18.32	0.3561
	Quantization indexing ($Q = 50$)	32.47	0.4576
	Quantization indexing ($Q = 100$)	46.74	0.9802

model in-use. In most case of pre-trained models, the proposed quantization indexing method ($Q = 100$) can easily outperform LSH scheme in term of both retrieval accuracy and time. Details on both retrieval accuracy and retrieval time on the Oxford building dataset are presented in Table 2.

Finally, we benchmark our proposed indexing method with the $1M$ image MIRFLICKR retrieval dataset. From the experimental result, we can observe that retrieval accuracy of the brute-force search approach is still highest. However, the retrieval time of this method increases significantly as the size of the database is up to $1M$ images. With the average retrieval time of above 200 seconds, this method can not be applied in real-time application though its accuracy is highest. The LSH approach achieves much better retrieval time (about 3 seconds) but its accuracy is slightly lower when compared to the brute-force method. Our proposed indexing method ($Q = 100$) gives a great performance with an accuracy of about 45% mAP and retrieval time of about 0.9 second. The retrieval time of our method is much lower than the LSH scheme while its accuracy almost the same as LSH scheme. Details on both retrieval accuracy and retrieval time on the MIRFLICKR retrieval dataset are presented in Table 3.

To sum up, it is verified that our proposed indexing method is a novel approach. It can both gain good accuracy and improve the retrieval time regardless of image domains, methods to generate image embedding, and the dimension of the embedding vector. With an appropriate quantization factor Q, our method can easily outperform the LSH scheme in term of both retrieval accuracy and time, even when the size of the image database is at large scale.

6 CONCLUSIONS

In this paper, we define a new task in clothing retrieval system, which aims to minimize the retrieval time when user query for similar clothes in large database. Also, we propose a new architecture for fashion search system with state-of-the-art methods. Two new features including embedding object detection network on mobile device and quantization indexing are introduced to the system to address the mentioned problem. Through extensive experiments, it is verified that our proposed indexing methods can both reduce the retrieval time significantly and keep the retrieval accuracy at a high rate. With the proposed indexing method, our system are capable of finding multiple similar clothing items from large database at a short delay. Therefore, our system can well meet the requirements of real-time retrieval in real-world applications.

ACKNOWLEDGEMENT

This work is partially supported by ***Framgia Inc***. We would like to thank our colleagues at ***Framgia Inc*** for their advice and expertise. Without their support, this experiment wouldn't have been accomplished.

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: a system for large-scale machine learning.. In *OSDI*, Vol. 16. 265–283.

[2] Lukas Bossard, Matthias Dantone, Christian Leistner, Christian Wengert, Till Quack, and Luc Van Gool. 2012. Apparel classification with style. In *Asian conference on computer vision*. Springer, 321–335.

[3] Huizhong Chen, Andrew Gallagher, and Bernd Girod. 2012. Describing clothing by semantic attributes. In *European conference on computer vision*. Springer, 609–623.

[4] Mayur Datar, Nicole Immorlica, Piotr Indyk, and Vahab S Mirrokni. 2004. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the twentieth annual symposium on Computational geometry*. ACM, 253–262.

[5] Manda Sai Divya and Shiv Kumar Goyal. 2013. ElasticSearch: An advanced and quick search technique to handle voluminous data. *Compusoft* 2, 6 (2013), 171.

[6] Ross Girshick. 2015. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*. 1440–1448.

[7] Yunchao Gong, Yangqing Jia, Thomas Leung, Alexander Toshev, and Sergey Ioffe. 2013. Deep convolutional ranking for multilabel image annotation. *arXiv preprint arXiv:1312.4894* (2013).

[8] M Hadi Kiapour, Xufeng Han, Svetlana Lazebnik, Alexander C Berg, and Tamara L Berg. 2015. Where to buy it: Matching street clothing photos in online shops. In *Proceedings of the IEEE international conference on computer vision*. 3343–3351.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[10] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861* (2017).

[11] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. 2017. Densely Connected Convolutional Networks.. In *CVPR*, Vol. 1. 3.

[12] Junshi Huang, Rogerio S Feris, Qiang Chen, and Shuicheng Yan. 2015. Cross-domain image retrieval with a dual attribute-aware ranking network. In *Proceedings of the IEEE international conference on computer vision*. 1062–1070.

[13] Mark J Huiskes and Michael S Lew. 2008. The MIR flickr retrieval evaluation. In *Proceedings of the 1st ACM international conference on Multimedia information retrieval*. ACM, 39–43.

[14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.

[15] Si Liu, Zheng Song, Guangcan Liu, Changsheng Xu, Hanqing Lu, and Shuicheng Yan. 2012. Street-to-shop: Cross-scenario clothing retrieval via parts alignment and auxiliary set. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 3330–3337.

[16] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. 2016. Ssd: Single shot multibox detector. In *European conference on computer vision*. Springer, 21–37.

[17] Ziwei Liu, Ping Luo, Shi Qiu, Xiaogang Wang, and Xiaoou Tang. 2016. Deepfashion: Powering robust clothes recognition and retrieval with rich annotations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1096–1104.

[18] James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. 2007. Object retrieval with large vocabularies and fast spatial matching. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*. IEEE, 1–8.

[19] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 779–788.

[20] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*. 91–99.

[21] Mohammad Javad Shafiee, Brendan Chywl, Francis Li, and Alexander Wong. 2017. Fast YOLO: A Fast You Only Look Once System for Real-time Embedded Object Detection in Video. *arXiv preprint arXiv:1709.05943* (2017).

[22] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[23] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1–9.

[24] Jingdong Wang, Heng Tao Shen, Jingkuan Song, and Jianqiu Ji. 2014. Hashing for similarity search: A survey. *arXiv preprint arXiv:1408.2927* (2014).

[25] Jiang Wang, Yang Song, Thomas Leung, Chuck Rosenberg, Jingbin Wang, James Philbin, Bo Chen, and Ying Wu. 2014. Learning fine-grained image similarity with deep ranking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1386–1393.

[26] Di Wei, Wah Catherine, Bhardwaj Anurag, Pira-muthu Robinson, and Sundaresan Neel. 2013. Style finder: fine-grained clothing style recognition and retrieval. In *Computer Vision and Pattern Recognition Workshops*. 8–13.

[27] Kota Yamaguchi, M Hadi Kiapour, Luis E Ortiz, and Tamara L Berg. 2012. Parsing clothing in fashion photographs. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 3570–3577.