Deep Metric Learning for Multi-Label and Multi-Object Image Retrieval

Jonathan MOJOO\textsuperscript{a)}, Nonmember and Takio KURITA\textsuperscript{††, Member}

SUMMARY  Content-based image retrieval has been a hot topic among computer vision researchers for a long time. There have been many advances over the years, one of the recent ones being deep metric learning, inspired by the success of deep neural networks in many machine learning tasks. The goal of metric learning is to extract good high-level features from image pixel data using neural networks. These features provide useful abstractions, which can enable algorithms to perform visual comparison between images with human-like accuracy. To learn these features, supervised learning of image similarity or relative similarity is often used. One important issue in deep metric learning is how to define similarity for multi-label or multi-object scenes in images. Traditionally, pairwise similarity is defined based on the presence of a single common label between two images. However, this definition is very coarse and not suitable for multi-label or multi-object data. Another common mistake is to completely ignore the multiplicity of objects in images, hence ignoring the multi-object facet of certain types of datasets. In our work, we propose an approach for learning deep image representations based on the relative similarity of both multi-label and multi-object image data. We introduce an intuitive and effective similarity metric based on the Jaccard similarity coefficient, which is equivalent to the intersection over union of two label sets. Hence we treat similarity as a continuous, as opposed to discrete quantity. We incorporate this similarity metric into a triplet loss with an adaptive margin, and achieve good mean average precision on image retrieval tasks. We further show, using a recently proposed quantization method, that the resulting deep feature can be quantized whilst preserving similarity. We also show that our proposed similarity metric performs better for multi-object images than a previously proposed cosine similarity-based metric. Our proposed method outperforms several state-of-the-art methods on two benchmark datasets.

key words: multi-label retrieval, deep metric learning, soft similarity, deep hashing

1. Introduction

Image retrieval is one of the popular areas of interest in computer vision and pattern recognition today. Research in this area began in the early 1990’s [1], [2] and recently, with the increase in the number of digital images collected and used in various domains, there is need for more accurate and faster algorithms for searching and retrieving images from large collections. Traditionally, real-world image retrieval systems use meta-data information associated with images to index and retrieve images. However, since it has been observed that textual information may be inconsistent with visual content, content-based image retrieval (CBIR) is preferred [3]. CBIR is the search and retrieval of images from large-scale image databases using visual features such as texture, shape and color [4].

One of the challenges in CBIR remains the ‘semantic gap’ [5], which refers to the difficulty of describing high-level semantic concepts that humans are familiar with using low-level visual features [6]–[8]. Bridging this gap requires understanding and comparison of image content at a semantic level similar to that demonstrated by humans [6].

The ability to assess how closely one image matches another is an indispensable component of image retrieval systems, especially in search-by-example applications, where the aim is to retrieve images that are similar to a supplied query image. It has been shown that features extracted by convolutional neural networks can provide useful representations of image data [9], [10], which can in turn be used for various image processing and recognition tasks [11]. In deep metric learning [12], the goal is to learn such representations and their corresponding induced metrics by training the network to estimate similarity or relative similarity of image samples. The learned embeddings and their corresponding similarity metric can then be used directly in image retrieval, usually by performing a nearest neighbour search.

It has been argued that the ability to classify local image regions into semantic object or concept classes e.g. water, rocks, person is key to achieving semantic scene understanding which is essential for better CBIR performance [13]. Furthermore, when considering semantic image retrieval, it is more desirable to represent and interpret images as scenes (that is, a collection of interacting objects and concepts) as opposed to collections of isolated objects [14].

In line with the above, our main contributions include the following:

1. A triplet metric learning architecture that combines the feature aggregation power of multi-layer perceptrons with the object-relation encoding and enumeration capabilities of relation networks, to learn embeddings that can be used to reliably compare image scenes.
2. A novel soft-similarity function for computing the similarity of multi-label image data. We show that our soft-similarity leads to better performance than a previously proposed approach, and works for both multi-label and multi-object images.
3. An adaptive margin triplet loss function that uses a similarity coefficient to adapt the margin for varying degrees of similarity.

Furthermore, we show that the embeddings learned by our architecture are quantizable, by employing a previously proposed quantization method that works on top of a triplet metric learning network.

2. Related Work

2.1 Deep Metric Learning

In simplest terms, the metric learning problem is as follows: given an initial distance function \(d(x, y)\) between two points \(x\) and \(y\) and a teacher signal representing the ground truth, construct a new distance function \(d'(x, y)\) that is closer to the ground truth distance than the original function. Note that this definition does not strictly require a distance function i.e. it holds for any similarity function \(s\). In deep metric learning, the problem becomes one of constructing a distance function \(d'(f(x), f(y))\) where \(f\) is a deep neural network that extracts features \(f(x)\) and \(f(y)\) given inputs \(x\) and \(y\) respectively. There are two predominant models: Siamese networks [15]–[17] and Triplet networks [12], [18]–[20], which focus on contrastive embedding and triplet embedding respectively.

A basic Siamese network consists of a two-branch network that learns contrastive embedding from pairs of samples. The two parallel branches share parameters, and the model learns by minimizing a cost function that favors a small distance between pairs of samples labeled as similar and a large distance between pairs labeled as dissimilar. One short fall of this type of model is that it requires training data with real-value precise pairwise distances which is not usually available [21]. It has also been observed that representations learned by these models provide sub-par results and for some types of data, the model almost completely fails to learn [12].

The Triplet network model is perhaps one of the most popular deep metric learning models [12], [18]–[20]. This model learns a triplet embedding by exploring the relative similarity of image samples. A triplet is made up of three samples: an anchor, a positive sample that is similar to the anchor and a negative sample that belongs to a different category from the anchor. The network learns by minimizing a cost function which favors an anchor-positive distance that is smaller than the anchor-negative distance. There have been multiple variations to this function [22], [23], but primarily it is defined as:

\[
L(a, p, n) = \sum_{i=1}^{N} \max\{d(a_i, p_i) - d(a_i, n_i) + m, 0\} \tag{1}
\]

where \(d(x_i, y_i) = \|x_i - y_i\|^2\), \(a_i = f(I_i^a)\), \(p_i = f(I_i^p)\) and \(n_i = f(I_i^n)\) are the outputs of a network \(f\) for inputs \(I_i^a\), \(I_i^p\) and \(I_i^n\) respectively. \(I_i^a\), \(I_i^p\) and \(I_i^n\) are the anchor, positive and negative example from the \(i\)-th triplet. \(m\) is a parameter that controls the gap between the distances of the two image pairs, and \(N\) is the sample size. The Triplet network model has been widely used in deep metric learning and has achieved promising results [18], [24]. However, by using a fixed margin parameter, the cost function in Eq. (1) assumes equal ranking of all positive examples relative to the anchor. This function does not reflect the ground truth when images strongly exhibit a non-binary similarity ranking, and may in fact disturb training. We propose a simple but effective version of this function that varies the margin parameter based on degree of similarity.

2.2 Deep Hashing and Quantization

Deep representations are effective for accurate image retrieval but for large-scale image datasets, converting the representations into compact binary codes allows more efficient storage and retrieval.

The goal of deep hashing is to learn deep image representations and their binary codes while preserving similarity. Xia et al. [25] proposed an approach that splits the problem into two stages: learning similarity-preserving hash codes for training samples using a scalable coordinate descent method, and learning an image deep hashing function that fits the learned hash codes. Lai et al. [26] presented an approach that jointly learns deep features and a hashing function, inside a triplet formulation to enforce ranking.

Quantization methods, which represent each sample by a code that points to the nearest center in a codebook, have been shown to be superior to hashing methods in approximate nearest neighbour search. One of such methods, proposed by Cao et al. [27], learns a similarity-preserving, quantizable deep image representation by jointly minimizing a pairwise cosine loss and product quantization loss, respectively. Later, Cao et al. [28] adopted an approach that transforms deep image representations into a new space, where similarity with Word2Vec label embeddings is enforced using an adaptive margin loss, and quantization is achieved by minimizing quantization error of approximate inner-product search. An approach by Liu et al. [29] utilises a novel triplet selection approach for selecting hard triplets, called Group Hard, in conjunction with triplet quantization with weak orthogonality to reduce codebook redundancy. We adopt their quantization and triplet selection strategies in our approach.

2.3 Multi-Label Image Retrieval

Most deep metric learning and deep hashing methods define pairwise similarity coarsely on a single-label basis. That is to say, two images are considered similar if any one label matches and dissimilar if no labels match. However this definition fails to capture the similarity ranking of multi-object images, and falls short of the fine-grained scene comparison capabilities displayed by humans. Recently, several approaches have been proposed to address this issue. DSRH [30] uses a weighted ranking loss to constrain deep
hash codes to follow a ranking based on the number of common labels between data points. Another approach, presented in [31], first generates region proposals and then aggregates them into separate deep representations, which are combined and mapped to a single semantic hash code. IDHN [32] is another method targeted at multi-label images, that employs a soft-similarity based on the cosine distance of label vectors, and a mean square error function that constrains hash-codes to preserve this similarity.

To the best of our knowledge, all existing methods take into account the presence or absence of certain labels, but not their multiplicity. We propose a more generalized similarity function that is not only sensitive to the presence or absence of labels, but also the number of occurrences of their corresponding objects in the image. We also propose an architecture that is primed for extracting such information. For simplicity, we disregard examples where all labels are missing.

### 2.4 Relation Networks

A relation network (RN) [33] is a neural network module that is geared towards relational reasoning. An RN is defined simply by the composite function

\[
RN(O) = f_\phi \left( \sum_{i,j} g_\theta(o_i, o_j) \right)
\]

where \(O = \{o_1, o_2, \cdots, o_n\}\) is a set of objects, \(o_i\) is the \(i\)th object and \(f_\phi\) and \(g_\theta\) are functions with parameters \(\phi\) and \(\theta\) respectively [33]. The function \(g_\theta\) infers the ways in which two objects are related and \(f_\phi\) uses the sum of these relations to make a decision. The two functions \(g_\theta\) and \(f_\phi\) are feed forward neural networks and \(\theta\) and \(\phi\) are the weights of these networks. A CNN augmented with an RN was shown to offer superior performance in tasks involving object comparison and counting. A two-stage variant of the RN architecture was also used in [34] to extract relation-aware features and retrieve images with similar object relationships. In our work, we use an RN module in parallel with a fully connected layer to equip our model with the capability to individualise and count objects.

## 3. Proposed Method

Our proposed method is illustrated in Fig. 1. In this section, we describe the various components that make up our model.

### 3.1 Feature Extractor

Without loss of generality, we follow [27]–[29], [32] and adopt AlexNet [35] as our base CNN architecture. The feature extractor consists of five convolutional layers with a ReLU non-linearity after each layer and max-pooling after the first, second, and fifth convolutions. We use this architecture for simplicity and consistency with similar proposed methods, but other CNN architectures could also just as easily be used with our approach. Our features are the output of the fifth layer, consisting of 256 convolutional feature maps, max-pooled to a size of 6×6.

![An illustration of our architecture.](image_url)
3.2 Feature Aggregation

The features from the last convolutional layer are fed into a feature aggregation module that consists of a fully connected section and an RN in parallel. The fully connected layers combine all the locations from the incoming feature maps into a global, compact representation, whereas the RN encodes object relationships, and more importantly, instances. The fully connected section is made up of two layers from the AlexNet architecture, fc-6 and fc-7, and terminates in a linear layer that outputs a 64-dimensional vector, which we will denote $\mathbf{x}^{FC}$. The RN takes as its input $O$, the set of point-wise features from the last convolutional layer. It is a slight modification of Eq. (2), and is defined as:

$$\mathbf{x}^{RN} = RN(O) = f_{b}\left(\sum_{i,j} g_{a}(\mathbf{x}^{FC}, \mathbf{o}_{i}, \mathbf{o}_{j})\right)$$

(3)

where $\mathbf{x}^{FC}$ is appended to add global context to the processing of each pair. $g_{a}$ and $f_{b}$ consist of three fully connected layers each. We apply a ReLU activation after each layer except the last layer, which outputs a 64-dimensional vector, $\mathbf{x}^{RN}$. We apply a batch normalization layer before $f_{b}$, as we found that the summation in Eq. (3) leads to large gradients, which cause the model to diverge. The final compact deep representation is a concatenation of the outputs of the two parallel modules:

$$\mathbf{x} = [\mathbf{x}^{FC}, \mathbf{x}^{RN}]$$

(4)

where [·, ·] denotes concatenation.

3.3 Triplet Quantization

Out of the quantization schemes presented in Sect. 2.2, the one introduced in [29] most fits our approach, since it is targeted at the triplet embedding case. It also includes an orthogonality constraint, which prevents codeword duplication across codebooks. Furthermore, it was proven to yield better results than the product quantization presented in [27]. Therefore, we adopt the triplet quantization scheme presented in [29] with a slight modification. It involves learning a set of $M$ codebooks $C = [C_{1}, \ldots, C_{M}]$, with each codebook containing $K$ D-dimensional cluster-centroid vectors $C_{m} = [C_{m1}, \ldots, C_{mK}]$. Then for each deep representation, we learn a binary assignment vector $\mathbf{b}_{m}$, made up of $M$ indicator vectors where each indicator vector $b_{mi}$ selects 1 of the $K$ codewords in the $m$-th codebook to approximate the $i$-th deep representation $\mathbf{x}_{i}$. The set of codebooks $C$ is shared across all triplets to enable knowledge sharing. To reduce redundancy, an orthogonality penalty is applied across the $M$ codebooks. The entire quantization objective is defined as:

$$Q = \sum_{i=1}^{N} \left\| \mathbf{x}_{i} - \sum_{m=1}^{M} C_{m}b_{mi} \right\|_{2}^{2} + \gamma \sum_{m=1}^{M} \sum_{n=1}^{M} \left\| C_{m}C_{n} - I \right\|_{2}^{2}$$

(5)

where the second term, weighted by $\gamma$, is the orthogonality penalty. $N$ is the number of samples in a mini-batch, including anchors, positives, and negatives. Thus far, this is similar to the approach presented in [29]. In our case, we found that it was better to split the quantization of $\mathbf{x}^{FC}$ and $\mathbf{x}^{RN}$, as these branches learn different deep representations. We split the set of codebooks into two, where each part has $M/2$ codebooks and each codebook has $KD/2$-dimensional centroid vectors. Before splitting, the number of bits used for each code is $M \log_{2}K$. Assigning $M/2$ codebooks to each part of the output vector means the total number of bits remains $2(M/2) \log_{2}K = M \log_{2}K$. This allows us to preserve the number of bits used for each code, but requires the total number of codebooks $M$ to be even. Hence Eq. (5) becomes

$$Q = \sum_{i \in \{FC,RN\}} \left( \sum_{n=1}^{N} \left( \sum_{m=1}^{M/2} \left\| \mathbf{x}_{i} - \sum_{m=1}^{M/2} C_{m}b_{mi}^{*} \right\|_{2}^{2} + \gamma \sum_{m=1}^{M/2} \sum_{n=1}^{M/2} \left\| C_{m}^{*}C_{n}^{*} - I \right\|_{2}^{2} \right) \right)$$

(6)

During retrieval, we follow [27]–[29] and compute the Asymmetric Quantizer Distance (AQD), which is defined as the inner product similarity between the embedding of a given query and the vector obtained by reconstructing a database point $\mathbf{x}_{n}$ from its binary code. In our case, taking into account the splitting of the codebooks, it becomes:

$$AQD(q, x_{n}) = \sum_{i \in \{FC,RN\}} z_{q}^{*}\left( \sum_{m=1}^{M/2} C_{m}^{*}b_{mn}^{*} \right)$$

(7)

where $z_{q}$ is the embedding of query $q$. With AQD, we can pre-compute all possible values of the inner product $z_{q}^{*}C_{m}^{*}b_{mn}^{*}$ and store them in a query specific $M \times K$ lookup table from which we calculate the AQD between the query and each database point. This only requires $M$ table lookups and additions for each database point, which leads to a reduced computational cost of retrieval.

3.4 Adaptive Margin Triplet Loss

Given two label vectors $y_{i}$ and $y_{j}$, we compute a similarity measure based on the Jaccard similarity coefficient [36] as:

$$s_{ij} = \frac{\sum_{n=1}^{|L|} \min(y_{in}, y_{jn})}{\sum_{n=1}^{|L|} \max(y_{in}, y_{jn})}$$

(8)

where $|L|$ is the size of the label space under consideration and $y_{in} \geq 0$ is the number of occurrences of the $n$-th label for the $i$-th example. Put simply, this is the intersection over union (IOU) between two label vectors, which takes the number of occurrences of each object into account. This similarity measure has the nice property of being in the range $[0, 1]$, is robust against sparsity in label vectors, and treats both binary (for multi-label) and non-binary (for multi-object) label vectors. We modify the triplet loss function in Eq. (1) to adjust the margin based on the similarity of
the positive pair as follows

\[ L = \sum_{i=1}^{N} \max\{d(x^a_i, x^p_i) - d(x^a_i, x^n_i) + s_{ai,pi}, 0\} \]  

(9)

where \( s_{ai,pi} \) is the similarity between the anchor and positive from the \( i \)-th example. This is slightly similar to the function presented in [28]. However, there are some key differences: the loss presented in [28] is a function of cosine similarities with no constant part, whereas ours uses our proposed similarity measure to scale a constant margin, allowing us to use large margin values. Additionally, they consider cosine similarities between deep representations and word embeddings whereas we consider relative distances between deep representations. The final loss function is then defined as

\[ E = L + \lambda Q, \]  

(10)

where \( \lambda \) is a parameter that controls the strength of the quantization loss term.

4. Experiments

4.1 Datasets

We train our model on two benchmark datasets, NUS-WIDE [37], and MS-COCO [38]. NUS-WIDE contains 269,648 images, where each image is labeled with a subset of 81 concepts. We follow [27]–[29] and randomly sample 5000 images as queries, 10,000 images for training, and use the rest as training data. This dataset is an example of a multi-label image dataset. Two sample images are shown in Fig. 2.

MS-COCO contains 82,783 training images and 40,504 validation images, where objects in each image fall in one of 80 categories. We follow the procedure in [29] and randomly sample 5000 images as queries, 10,000 images for training, and use the rest as training data. This dataset is an example of a multi-object image dataset. We show two sample images from this dataset in Fig. 3.

4.2 Training

We adopt Group-Hard, a triplet selection scheme introduced in [29], with the number of groups \( |G| = 20 \) for MS-COCO and \( |G| = 50 \) for NUS-WIDE. Following [28], [29], we fix the number of codewords as \( K = 256 \). Thus for \( M \) codebooks, the binary code for each sample is \( 8M \)-bit long. We quantize the \( RN \) and \( FC \) components of the output vector separately, and assign an equal number of codebooks to each, so that the final number of bits becomes \( 2 \times 8M \).

During training, we fine-tune convolutional layers from conv-3 to conv-5, and fully connected layers fc-6 and fc-7 of AlexNet. The last fully connected layer, and the entire RN portion of the architecture are trained starting from a random Gaussian initialization. We use SGD with a momentum of 0.9 as our optimizer, and start with a learning rate of \( 10^{-5} \), later scaling it down by \( 10^{-1} \). We determined the rest of the parameters by doing parameter searches within reasonable ranges. We set \( m = 15 \) by doing a parameter search in the range \([5, 30]\), and \( \lambda = 0.001 \) by doing a parameter search in the range \([0, 1]\). We initially held back 1000 samples from the training set for validation and hyper-parameter tuning, then used the entire training set to train the final model.

5. Results and Analysis

Table 1 shows the MAP results of our method compared to some state-of-the-art methods, and it shows that our method outperforms the comparison methods, with margins as large as 5% on NUS-WIDE and 12.2% on MS-COCO. Specifically, our method shows gains of 0.4% and 4% over DTQ, a method which shares the most similarities with our method. This MAP is calculated using the coarse definition of similarity, that is, two data points are considered similar if they share at least 1 label, and dissimilar otherwise. This shows that for multi-label data, using the coarse definition of image similarity for triplet sampling hurts performance, even if we maintain the same definition during evaluation. We also note that our method outperforms the competing methods by a larger margin on the MS-COCO dataset than on the NUS-WIDE dataset. This proves that while our method offers competitive performance for multi-label data, there is

Table 1 | Mean average precision (MAP) for different number of bits on two benchmark datasets.

| Method     | NUS-WIDE 16 bits | NUS-WIDE 32 bits | MS-COCO 16 bits | MS-COCO 32 bits |
|------------|------------------|------------------|----------------|----------------|
| DQN [27]   | 0.735            | 0.752            | 0.653          | 0.685          |
| DVSQ [28]  | 0.790            | 0.797            | 0.712          | 0.720          |
| DTQ [29]   | 0.798            | 0.801            | 0.760          | 0.767          |
| MLMO (Ours)| 0.803            | 0.805            | 0.806          | 0.807          |

Fig. 2 | Two sample images from the NUS-WIDE dataset.

(a) sun, sunset, sky, clouds
(b) buildings, tower, nighttime, sky, clouds

Fig. 3 | Two sample images from the COCO dataset (the number in brackets is the number of occurrences of each object in the target image).

(a) person (2), frisbee (1)
(b) cow (8)
Table 2  Mean average precision (MAP) for our method and several of its variants.

| Method         | NUS-WIDE   |          |          | MS-COCO   |          |          |
|----------------|------------|----------|----------|-----------|----------|----------|
|                | deep feat  | 16 bits  | 32 bits  | deep feat | 16 bits  | 32 bits  |
| MLMO-RN        | 0.799      | 0.789    | 0.763    | 0.775     | 0.773    | 0.774    |
| MLMO-FC        | 0.799      | 0.797    | 0.799    | 0.799     | 0.794    | 0.795    |
| MLMO-cosine    | 0.800      | 0.796    | 0.800    | 0.773     | 0.774    | 0.775    |
| MLMO           | 0.805      | 0.803    | 0.805    | 0.806     | 0.806    | 0.807    |

Table 3  Mean average precision@0.50 (MAP@0.50) for our method and several of its variants on the NUS-WIDE dataset.

| Method         | NUS-WIDE   |          |          | MS-COCO   |          |          |
|----------------|------------|----------|----------|-----------|----------|----------|
|                | deep feat  | 16 bits  | 32 bits  | deep feat | 16 bits  | 32 bits  |
| DTQ [29]       | 0.398      | 0.392    | 0.400    | 0.121     | 0.117    | 0.116    |
| MLMO-RN        | 0.454      | 0.366    | 0.354    | 0.123     | 0.119    | 0.170    |
| MLMO-FC        | 0.451      | 0.449    | 0.447    | 0.173     | 0.170    | 0.170    |
| MLMO-cosine    | 0.455      | 0.444    | 0.447    | 0.148     | 0.145    | 0.146    |
| MLMO           | 0.454      | 0.449    | 0.451    | 0.173     | 0.168    | 0.170    |

Fig. 4  Retrieval examples for two variants of our method and the baseline method (using 32-bit quantized features).

marked improvement over previous methods when dealing with multi-object data.

Table 2 shows the MAP results for ablative experiments on our architecture. The results justify some architectural choices we made for our final model. MLMO-RN is a variant of our method using only the RN module after the feature extractor, whereas MLMO-FC is a variant using only the fully connected module. MLMO-cosine is a variant that uses cosine distance to compute similarity between label vectors, like some other works propose [28], [32]. Our final proposed model has the highest MAP for all numbers of bits on both datasets. It is worth noting that MLMO-FC,
version of our model with the RN module removed, still performs better than MLMO-cosine in multiple configurations (all configurations on MS-COCO, 16-bits on NUS-WIDE) and offers competitive performance in the rest of the configurations. This points to the importance of our IoU-based similarity measure.

Since the deep feature can also be directly used in nearest neighbour search, we include this result to show the information loss incurred due to the quantization step. We note that this is something other works on the same topic fail to include. Our results show no drop in MAP between the deep features and the 32-bit quantized features. This proves that our learned deep feature can be quantized while preserving similarity between images.

Table 3 shows MAP@0.50 results for the same ablative experiments shown in Table 2. This is similar to the standard MAP, but we only consider a retrieved image as positive if its similarity $s_{qi}$ with the query, defined in Eq. (8), is greater than 0.50. We also include results for the baseline, DTQ. We include this metric to investigate how well our model learns the objective based on the similarity measure we proposed, and how well it performs in multi-label and multi-object retrieval. Under each configuration, a variant of our method outperforms the baseline, which proves the superiority of our method on both multi-label and multi-object image retrieval. Except for the deep feature-based nearest neighbour search on NUS-WIDE, variants of our method using the proposed similarity metric give the best MAP@0.50. It is also worth noting that on the NUS-WIDE dataset, where label vectors are binary and there is no supervised information of object multiplicity, the margin between the models using cosine similarity and Jaccard index is small (0.4 under 32 bits%). On MS-COCO dataset, however, the margin is large (2.4%), emphasizing the superiority of our proposed similarity metric for multi-object retrieval.

Figure 4 shows the top-10 retrieved results for our method, its cosine variant, and the baseline method. The IOU between the label vectors of the retrieved image and the query is shown against each result. All the examples use 32-bit quantized features. For the NUS-WIDE example, the cosine and Jaccard similarity variants of our model offer comparable performance, with our approach (MLMO) returning slightly more results with an IOU over 0.5. Both variants perform better than the baseline. The MS-COCO example shows that our method is better able to factor in object multiplicity during retrieval than both the cosine similarity-based variant and the baseline.

6. Conclusion

In this paper, we introduced a deep metric learning approach focused on scene comparison between multi-label and multi-object images. We define a simple and effective similarity measure that enables our model to learn similarity preserving deep representations for multi-label and multi-object images. We show that our method offers better performance than some state-of-the-art methods on two benchmark datasets. Our work represents an important step towards fine-grained scene comparison in CBIR. In future, we would like to introduce a similarity measure that also takes into account the relations between objects in an image, and an architecture that reliably preserves the similarity of entire scene graphs.

References

[1] T. Kato, T. Kurita, and H. Shimogaki, “Multimedia interaction with image database systems,” ACM SIGCHI Bulletin, vol.22, no.1, pp.52–54, 1990.
[2] Y. Rui, T.S. Huang, M. Ortega, and S. Mehrotra, “Relevance feedback: a power tool for interactive content-based image retrieval,” IEEE Transactions on Circuits and Systems for Video Technology, vol.8, no.5, pp.644–655, 1998.
[3] T. Takahashi and T. Kurita, “Mixture of subspaces image representation and compact coding for large-scale image retrieval,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.37, no.7, pp.1469–1479, 2015.
[4] A.K. Alhassan and A.A. Alfaki, “Color and texture fusion-based method for content-based image retrieval,” Proc. 2017 International Conference on Communication, Control, Computing and Electronics Engineering, pp.1–6, IEEE, 2017.
[5] P.P. Mane and N.G. Bawane, “An effective technique for the content based image retrieval to reduce the semantic gap based on an optimal classifier technique,” Pattern Recognition and Image Analysis, vol.26, no.3, pp.597–607, 2016.
[6] A. Alzu’bi, A. Amira, and N. Ramzan, “Semantic content-based image retrieval: A comprehensive study,” Journal of Visual Communication and Image Representation, vol.32, no.Supplement C, pp.20–54, 2015.
[7] T. Kurita and T. Kato, “Learning of personal visual impression for image database systems,” Proc. 2nd International Conference on Document Analysis and Recognition, pp.547–552, 1993.
[8] X. Li, T. Uricchio, L. Ballan, M. Bertini, C.G.M. Snoek, and A.D. Bimbo, “Socializing the semantic gap: A comparative survey on image tag assignment, refinement, and retrieval,” ACM Comput. Surv., vol.49, no.1, pp.14:1–14:39, 2016.
[9] M.D. Zeiler and R. Fergus, “Stochastic pooling for regularization of deep convolutional neural networks,” 2013, URL https://arxiv.org/abs/1301.3557.
[10] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, “Overfeat: Integrated recognition, localization and detection using convolutional networks,” 2013, URL https://arxiv.org/abs/1312.6229.
[11] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, “Cnn features off-the-shelf: an astounding baseline for recognition,” Proc. IEEE conference on computer vision and pattern recognition workshops, pp.806–813, 2014.
[12] E. Hoffer and N. Ailon, “Deep metric learning using triplet network,” Proc. International Workshop on Similarity-Based Pattern Recognition, vol.9370, pp.84–92, Springer, 2015.
[13] J. Vogel and B. Schiele, “Semantic modeling of natural scenes for content-based image retrieval,” International Journal of Computer Vision, vol.72, no.2, pp.133–157, 2007.
[14] J. Johnson, R. Krishna, M. Stark, L.-J. Li, D.A. Shamma, M.S. Bernstein, and L. Fei-Fei, “Image retrieval using scene graphs,” Proc. 2015 IEEE Conference on Computer Vision and Pattern Recognition, pp.3668–3678, 2015.
[15] X. Han, T. Leung, Y. Jia, R. Sukthankar, and A.C. Berg, “Matchnet: Unifying feature and metric learning for patch-based matching,” Proc. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.3279–3286, 2015.
[16] J. Masci, D. Migliore, M.M. Bronstein, and J. Schmidhuber, “Descriptor learning for omnidirectional image matching,” Proc. Regis-
