Attribute Conjunction Learning with Recurrent Neural Network

Kongming Liang[1,2], Hong Chang[1(✉)], Shiguang Shan[1], and Xilin Chen[1,2]

[1] Key Lab of Intelligent Information Processing
of Chinese Academy of Sciences (CAS), Institute of Computing Technology, CAS,
Beijing 100190, China
kongming.liang@vipl.ict.ac.cn, {changhong,sgshan,xlchen}@ict.ac.cn
[2] University of Chinese Academy of Sciences, Beijing 100049, China

Abstract. Searching images with multi-attribute queries shows practical significance in various real world applications. The key problem in this task is how to effectively and efficiently learn from the conjunction of query attributes. In this paper, we propose Attribute Conjunction Recurrent Neural Network (AC-RNN) to tackle this problem. Attributes involved in a query are mapped into the hidden units and combined in a recurrent way to generate the representation of the attribute conjunction, which is then used to compute a multi-attribute classifier as the output. To mitigate the data imbalance problem of multi-attribute queries, we propose a data weighting procedure in attribute conjunction learning with small positive samples. We also discuss on the influence of attribute order in a query and present two methods based on attention mechanism and ensemble learning respectively to further boost the performance. Experimental results on aPASCAL, ImageNet Attributes and LFWA datasets show that our method consistently and significantly outperforms the other comparison methods on all types of queries. The software related to this paper is available at https://github.com/GriffinLiang/AC-RNN.

Keywords: Attribute learning · Multi-label learning · Image retrieval

1 Introduction

Attribute learning provides a promising way for computer to understand image content in a fine-grained manner. Beyond traditional object categories, attribute contains abundant information from holistic perception (e.g., color, shape, etc.) to the presence or absence of local parts for images. By bridging the gap between low-level features and high-level categorization, attributes benefit many object recognition and classification problems (e.g., object recognition [26], face verification [15] and zero-shot classification [16]).

Compared with single attribute learning, learning attribute conjunctions shows more practical significance. An attractive application is to retrieve relevant images based on multi-attribute query. For example, it can be used to discover objects with

P. Frasconi et al. (Eds.): ECML PKDD 2016, Part I, LNAI 9851, pp. 345–360, 2016.
DOI: 10.1007/978-3-319-46128-1_22

specified characteristics [5,22], search people of certain facial descriptions [14] and match products according to users' requirements [13]. In this scenario, a user may describe the visual content of interest by specifying a few attributes. Then the image search engine will calculate the similarity between the input attribute conjunction and the images in some datasets. The most similar images will be returned as the search results.

A common approach [5,16] to tackle multi-attribute query is transforming the problem into multiple single-attribute learning tasks. Specifically, a binary classifier is built for each single attribute, then the result of multi-attribute prediction is generated by summing up the scores of all single attribute classifiers. Though this kind of combination is simple and shows good scalability, it has two main drawbacks. Firstly, the correlation between attributes is ignored because of the separate training of each attribute classifier. Secondly, attribute classification results are sometimes unreliable since abstract linguistic properties can have very diverse visual manifestations especially when they come across different categories. This situation may get worse with larger number of attributes appearing in a query. For example, when the query length is three, an unreliable attribute classifier may affect $\binom{A-1}{2}$ query results (A is the total number of attributes).

Instead of training a classifier for each attribute separately, a more promising approach is to learn from the attribute conjunctions. Since conjunctions of multiple attributes may lead to very characteristic appearances, training a classifier that detects the conjunctions as a whole may produce more accurate results. For example, training a classifier to predict whether the animal is (black & white & stripe) leads to a specific concept "Zebra". However, straightforward training classifiers from attribute conjunctions is not a good choice. Firstly, the length of multi-attribute query is not fixed and the number of attribute conjunctions grows exponentially w.r.t. the query length. For a three-attribute query, we need to build $\binom{A}{3}$ classifiers for all possible attribute conjunctions. Secondly, there are only a small number of positive examples for each multi-attribute query (a positive sample must have multiple query attributes simultaneously), which brings the learning process more difficulties. With data bias problem, some attribute conjunction classifiers may perform even worse than simply adding the scores of disjoint single-attribute classifiers. Thirdly, the correlation between attribute conjunctions is not well explored, since the queries which share common attributes are considered to be independent from each other.

In this paper, we propose a novel attribute conjunction recurrent neural network (AC-RNN) to tackle multi-attribute based image retrieval problem. As shown in Fig. 1, the input sequence of AC-RNN are the attributes appearing in a query with a predefined order. Each of the input attributes is then embedded into the hidden units and combined in a recurrent way to generate the representation for the attribute conjunction. The conjunction representation is further used to compute the classifier for the input multi-attribute query. As the multiple attributes in each conjunction are processed by the network recurrently, the number of parameters of our model do not increase with the length

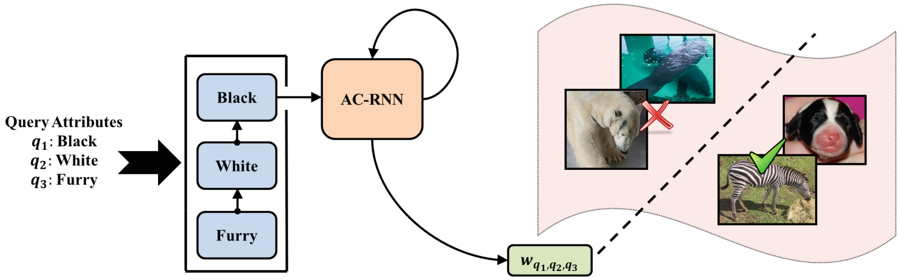


Fig. 1. An Illustration of Attribute Conjunction Recurrent Neural Network (AC-RNN).

of query. Compared with straightforward multi-attribute learning methods, our AC-RNN model is more appropriate to model the complex relationship among different attribute conjunctions. We also introduce a data weighting procedure to address the data bias problem in attribute conjunction learning. Finally, we discuss on the influence of attribute order in our learning framework and propose two methods based on attention mechanism and ensemble learning respectively to improve the performance of AC-RNN.

The rest of this paper is organized as follows. We first introduce some related works in the following section. In Sect. 3, we present the attribute conjunction recurrent neural network in detail. Experimental results are then shown in Sect. 4. Finally, Sect. 5 concludes the paper.

2 Related Work

Multi-Attribute Query[1]: Guillaumin et al. [8] propose to use a weighted nearest-neighbour model to predict the tags of a test image which can directly support multi-word query based image retrieval. Petterson et al. [19] present a reverse formulation to retrieve sets of images by considering labels as input. In this way, they can directly optimize the convex relaxations of many popular performance measures. By leveraging the dependencies between multiple attributes, Siddiquie et al. [23] explicitly model the correlation between query-attributes and non-query attributes. For example, for a query such as "man with sunglasses", the correlated attributes like beard and mustache can also be used to retrieve releant images. Since training a classifier for a combination of query attributes may not always perform well, Rastegari et al. [20] propose an approach to determine whether to merge or split attributes based on the geometric quantities. Different from the above methods, we propose to explicitly learn all the single attribute embeddings and combine them in a recurrent way to generate the representation of attribute conjunction.

[1] The multi-attribute we denote here may refer to other statements such as keywords or multi-label in other literature.

Multi-label Learning: Read et al. [21] extend traditional binary relevance method by predicting multiple attribute progressively in an arbitrary order. For instance, one label is predicted first. Then the prediction result is appended at the end of the input feature which is used as the new feature to predict the second label. Finally, the multiple label predictions form into a classifier chain. Since a single standalone classifier chain model can be poorly ordered, the authors also propose an ensemble method in a vote scheme. Zhang et al. [29] exploit different feature set to benefit the discrimination of multiple labels. This method exploits conducting clustering analysis on the positive and negative instances and then performs training and testing referring to the clustering results. Different from multi-label learning problems, the task we deal with here is to model attribute conjunctions instead of multiple separate labels.

Label Embedding: Though deep learning provides a powerful way to learn data representations, how to represent labels is also a key issue for machine learning methods. A common way is Canonical Correlation Analysis (CCA) [9] which maximizes the correlation between data and labels by projecting them into a common space. Another promising way is to learn label embedding by leveraging other possible sources as prior information. Akata et al. [1] propose to embed category labels into attribute space under the assumption that attributes are shared across categories. Frome et al. [6] represent category labels with the embedding learned from textual data. Hwang et al. [11] jointly embed all semantic entities including attributes and super-categories into the same space by exploiting taxonomy information. But so far, there is no work on learning the conjunction representation of multiple labels to the best of our knowledge.

3 Our Method

The problem we aim to address here is to retrieve relevant images according to the user's query. Intuitively, multi-attribute queries are conjunctions of single attributes and the correlation between them is usually strong. Therefore it is critical to learn from all attribute conjunctions jointly. Firstly, we propose to use the recurrent neural network to model complex attribute conjunctions. The model can not only reveal the representation of attribute conjunctions but also output the multi-attribute classifiers. Secondly, we integrate the generated classifiers into traditional logistic regression model. The parameters of recurrent neural network and logistic regression are optimized simultaneously using back propagation. We also propose a weighted version of our model to tackle data imbalance problem. Thirdly, we study the influence of attribute order in each query and propose two methods to further improve the original formulation.

3.1 Attribute Conjunction Learning

Let $\mathcal{Q} = \{\mathbf{Q}^1, \mathbf{Q}^2, ..., \mathbf{Q}^M\}$ be a set of M multi-attribute queries. The m^{th} query is represented as a matrix $\mathbf{Q}^m = (\mathbf{q}_1^m, \mathbf{q}_2^m, ..., \mathbf{q}_{T_m}^m) \in \{0,1\}^{A \times T_m}$, where A is the number of predefined attributes and T_m is the number of attributes appearing in the m^{th} query. $\mathbf{q}_t^m = [\mathbf{q}_{1t}^m, \dots, \mathbf{q}_{At}^m]^T$ is a one-hot query vector, where $q_{at}^m = 1$ if the t^{th} attribute in the current query is attribute a and $q_{at}^m = 0$ otherwise.

Our model takes multi-attribute query as input and outputs the representation of the query as well as the multi-attribute classifiers. More specifically, for the m^{th} multi-attribute query, the one-hot query vectors $\mathbf{q}_t^m$ $(t = 1, \dots, T_m)$ corresponding to the attributes involved in the query are input sequentially to our model. The subscript t decides the input order. We learn the multi-attribute conjunction in a recurrent way, as illustrated in Fig. 2. In this model, the first t attributes of the m^{th} query can be represented as:

$$\begin{aligned} \mathbf{h}_t^m &= f_h(\mathbf{q}_t^m, \mathbf{h}_{t-1}^m) \\ &= \sigma(\mathbf{W}_v \mathbf{q}_t^m + \mathbf{W}_h \mathbf{h}_{t-1}^m + \mathbf{b}_h), \end{aligned} \tag{1}$$

where f_h is a *conjunction function* to model the relationship of all the attributes belonging to the m^{th} query. $\mathbf{W}_v \in \mathbb{R}^{H \times A}$ and $\mathbf{W}_h \in \mathbb{R}^{H \times H}$ are *embedding* and *conjunction matrix* respectively, where H is the number of hidden units of the recurrent network. $\mathbf{h}_0^m \equiv \mathbf{h}_0$ represents the initial hidden state. $\mathbf{b}_h$ is the bias and $\sigma(\cdot)$ is an element-wise non-linear function which is chosen to be sigmoid in this paper.

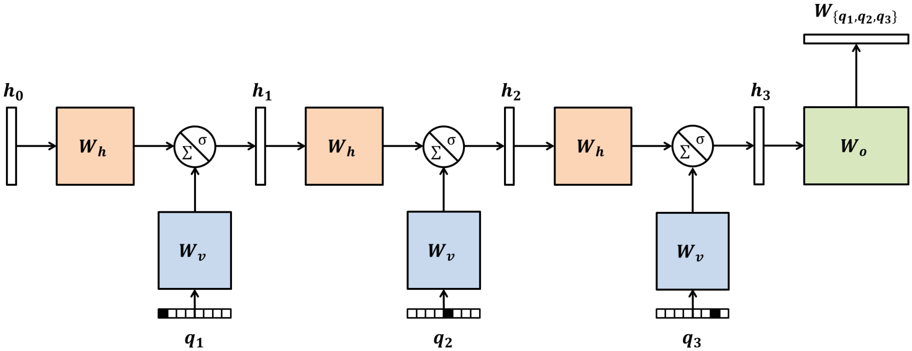


Fig. 2. An Illustration of Attribute Conjunction Recurrent Neural Network (AC-RNN) with triple attribute query.

From Eq. (1), we can see that each column of parameter matrix $\mathbf{W}_v$ can be considered as single attribute representation, noting that the query vector is in one-hot form. Therefore, all input queries, including long queries with many user specified attributes, share the same attribute-level representations. In this way, the parameter growth problem for long queries is addressed.

After computing $\mathbf{h}_{T_m}^m$ of the last query vector with the recurrent network, we actually obtain the hidden representation of the whole query. Then, we stack one layer on top of the recurrent units and compute the *multi-attribute classifier* $\mathbf{w}_m$ as the output of the neural network:

$$\mathbf{w}_m = f_o(\mathbf{h}_{T_m}^m) = \mathbf{W}_o \mathbf{h}_{T_m}^m + \mathbf{b}_o. \tag{2}$$

Here, the regression function f_o is chosen to be in a linear form, though more complex form can be considered. The parameter $\mathbf{W}_o$ and $\mathbf{b}_o$ are the output matrix and bias respectively. In this way, the attribute embeddings of the current query are combined in a recurrent way to learn the complex relationship between the attributes. After that we use the output conjunction as the m^{th} multi-attribute classifier to retrieve relevant images. The model parameters of the conjunction and output functions are denoted as $\Theta = \{\mathbf{W}_v, \mathbf{W}_h, \mathbf{W}_o, \mathbf{b}_h, \mathbf{b}_o, \mathbf{h}_0\}$.

3.2 Multi-attribute Classification

Suppose there are N labeled images, $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^N$, where $\mathbf{x}_i \in \mathbb{R}^D$ denotes the D-dimensional image feature vector, and $\mathbf{y}_i \in \{0,1\}^A$ indicates the absence and presence of all attributes. The attribute label can be expressed in matrix form as $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_N] \in \{0,1\}^{A \times N}$. In order to retrieve relevant images given a multi-attribute query $\mathbf{Q}^m$, we resort to multi-attribute classification to estimate the labels $\mathbf{Y}$.

Since attribute learning is a binary classification problem, we make use of logistic regression to predict the absence or presence of multiple attributes. The loss function with respect to the m^{th} multi-attribute query is expressed as the following negative log likelihood:

$$\begin{aligned} L(\mathbf{x}_i, \mathbf{y}_i, \mathbf{Q}^m; \Theta) = &-\tilde{\mathbf{y}}_{im} log(\sigma(\mathbf{w}_m^T \mathbf{x}_i)) \\ &-(1 - \tilde{\mathbf{y}}_{im}) log(1 - \sigma(\mathbf{w}_m^T \mathbf{x}_i)), \end{aligned} \tag{3}$$

where $\tilde{\mathbf{y}}_{im} = (\mathbf{y}_i^T \mathbf{q}_1^m \ \& \ \mathbf{y}_i^T \mathbf{q}_2^m \ \& \ \cdots \ \& \ \mathbf{y}_i^T \mathbf{q}_{T_m}^m)$ and & denotes the bitwise operation AND . $\mathbf{w}_m$ is the multi-attribute classifier computed from Eq. (2).

Generally speaking, the presences of some attributes are usually much less than its absence. This situation is even worse for multi-attribute image retrieval since the positive sample must have multiple query attributes simultaneously. To tackle the sample imbalance problem, we evolve our formulation with *data weighting* procedure inspired by [8,12]. The resulting loss function is rewritten as the following weighted log likelihood:

$$\begin{aligned} L_w(\mathbf{x}_i, \mathbf{y}_i, \mathbf{Q}^m; \Theta) = &-c_m^+ \tilde{\mathbf{y}}_{im} log(\sigma(\mathbf{w}_m^T \mathbf{x}_i)) \\ &-c_m^-(1 - \tilde{\mathbf{y}}_{im}) log(1 - \sigma(\mathbf{w}_m^T \mathbf{x}_i)), \end{aligned} \tag{4}$$

where $c_m^+ = N/(2 \times N_m^+)$ and $c_m^- = N/(2 \times N_m^-)$ which make the loss weights of all the data sum up to N. N_m^+ (N_m^-) is the number of positive (negative) images for the m^{th} multi-attribute query. The experimental results show that the weighted loss function performs better than the original logistic regression.

By combining the attribute conjunction and multi-attribute classification into a unified framework, the final objective function is formulated in the following form:

$$\arg\min_{\Theta} \frac{1}{N} \sum_{i=1}^{N} \sum_{m=1}^{M} L_*(x_i, y_i, Q^m; \Theta) + \lambda \Omega(\Theta), \tag{5}$$

where $\Omega(\cdot)$ is the weight decay term used to increase the model generalization ability. The parameters λ is used to balance the relative influence of the regularization terms. The loss function can also be replaced with the weighted form as defined in Eq. (4).

We solve the above optimization problem by using L-BFGS. The derivatives of the logistic regression parameters are calculated and back propagated into the output units of AC-RNN. Then the derivatives of Θ can be easily computed with the backpropagation through time algorithm [27]. In this way, our model can be trained in an end-to-end manner.

3.3 Attribute Order in AC-RNN

Recurrent neural networks are well suited to model sequential data. However, the input query attributes are not naturally organized as a sequence since the underlying conditional dependency between attributes are not known. The performance of our model is somewhat sensitive to the input order of attributes. To tackle this problem, we propose two methods by using recurrent attention mechanism and ensemble learning respectively.

Attention Mechanism Based. Attention mechanism has been successfully applied in generating image caption [28], handwriting recognition [7] and machine translation [2]. And it have been used to model the input and output structure of a sequence to sequence framework in a recent paper [25]. Inspired by the previous works, we propose to integrate the attention mechanism into our model to tackle the ordering problem. The pipeline is shown in Fig. 3. The proposed network reads all the attributes according to an attention vector, instead of processing query attributes one by one at each step. The attention vector is a probability vector indicating the relevance of all pre-defined attributes to the current query. And it is automatically modified at each processing step and recurrently contributes to the representation of the attribute conjunction. Intuitively, we first initialize the input attention vector for the m^{th} query as:

$$\mathbf{p}_1^m = \frac{\sum_{i=1}^{T_m} \mathbf{q}_i^m}{T_m}. \tag{6}$$

In this way, the network will first take the query attributes into attention. Then we refined the attention vector and learn the attribute conjunction step by step using the recurrent neural network with attention mechanism. In the t^{th} step, the attribute conjunction and attention vector are generated as follows:

$$\mathbf{h}_t^m = f_h(\mathbf{p}_t^m, \mathbf{h}_{t-1}^m) = \sigma(\mathbf{W}_v \mathbf{p}_t^m + \mathbf{W}_h \mathbf{h}_{t-1}^m + \mathbf{b}_h), \tag{7}$$

$$\mathbf{p}_{t+1}^m = Softmax(\mathbf{U}\mathbf{h}_t^m) = \frac{1}{\sum_{j=1}^{A} e^{\mathbf{U}_j^{\mathrm{T}}\mathbf{h}_t^m}} \begin{bmatrix} e^{\mathbf{U}_1^{\mathrm{T}}\mathbf{h}_t^m} \\ e^{\mathbf{U}_2^{\mathrm{T}}\mathbf{h}_t^m} \\ \vdots \\ e^{\mathbf{U}_A^{\mathrm{T}}\mathbf{h}_t^m} \end{bmatrix}, \tag{8}$$

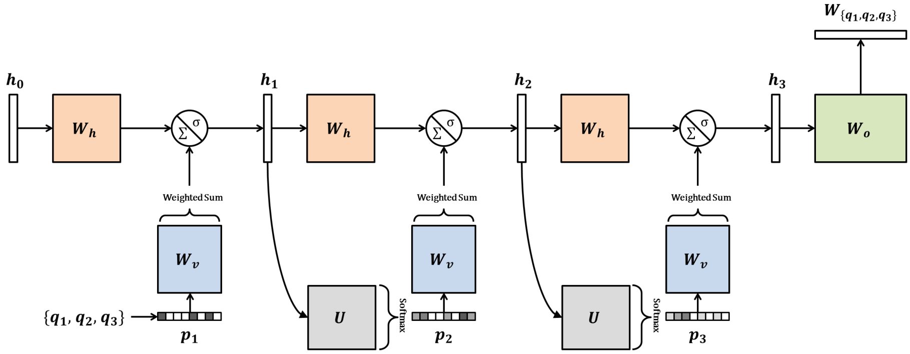


Fig. 3. AC-RNN with attention mechanism (AC-RNN-ATN).

where the *attention matrix* $\mathbf{U} \in \mathbb{R}^{H \times A}$ transforms the hidden units into the attention vector of the next processing step. The other parameters are consistent with the definition in Sect. 3.1.

By using a recurrent attention model, the output attribute conjunction is invariant to the input order. In addition, by taking the non-query attributes into consideration, this model can leverage the co-occurrence information to enhance the query attribute conjunction. Therefore, an unreliable query attribute might piggyback on an co-occurring attribute that has abundant training data and easier to predict.

Ensemble Based. Another method to alleviate the influence of attribute order is directly using the ensemble of original models. Therefore, we present Ensembles of AC-RNN to reduce the negative effect of poorly ordered input. The ensemble framework can be trained in parallel without increasing the overall time cost. Instead of using the pre-defined attribute order, a random order of all the attributes is generated for each model. Then the input attributes are rearranged according to the generated order for each multi-attribute query (Fig. 4).

The model parameters of each AC-RNN are learned to obtain a multi-attribute classifier. At the last stage, the outputs of all the independent models are averaged to obtain the final multi-attribute classifier:

$$\mathbf{w}_m = \frac{1}{C} \sum_{c=1}^{C} \mathbf{w}_m^c, \tag{9}$$

where C is the number of models in the ensemble and $\mathbf{w}_m^c$ represents the weight of multi-attribute classifier generated by the c^{th} model. The ensemble of multi-attribute classifier is further used to retrieve the relevant images.

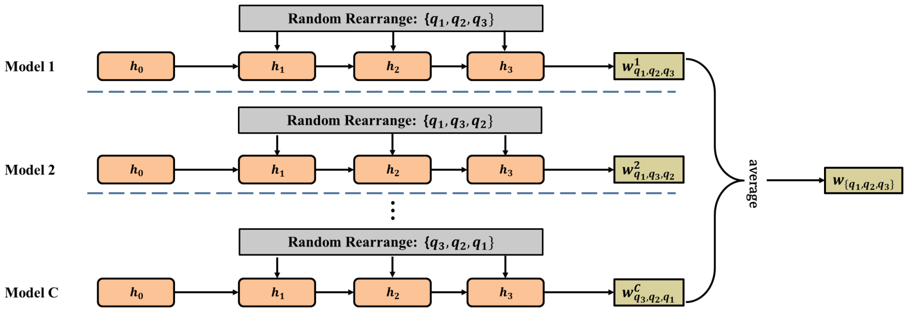


Fig. 4. AC-RNN based on ensemble learning.

4 Experiments

We evaluate our method[2] on three widely used datasets: aPascal [5], ImageNet Attributes [22] and LFWA [17]. Then we verify the effectiveness of the weighted version and visualize the ground truth correlation matrix and the learned similarity matrix for comparison. Finally, we present the experimental results of the proposed two methods to evaluate the influence of attribute order.

4.1 Datasets

aPASCAL. This dataset contains 6430 training images and 6355 testing images from Pascal VOC 2008 challenge. Each image comes from twenty object categories and annotated with 64 binary attribute labels. We use the pre-defined test images for testing and randomly split ten percent images from training set for validation. The feature we used for all the comparison methods are called DeCAF [4] which are extracted by the Convolutional Neural Networks (CNN). Since attributes are only defined for objects instead of the entire image, we use the object bounding box as the input of CNN.

ImageNet Attributes (INA). ImageNet Attribute dataset contains 9,600 images from 384 categories. Each image is annotated with 25 attributes describing color, patterns, shape and texture. 3–4 workers are asked to provide a binary label indicating whether the object in the image contains the attribute or not. When there is no consensus among the workers, the attribute will be labeled as ambiguous for this image. The data with ambiguous attribute are not used for training and evaluating for the queries which contains the corresponding attribute. We use $\{60\,\%, 10\,\%, 30\,\%\}$ random split for training/validation/test. And we also use DeCAF to do feature extraction.

LFWA. The labelled images on this dataset are selected from the widely used face dataset LFW [10]. It contains 5,749 identities with totally 13,233

[2] The code of our method is available at https://github.com/GriffinLiang/AC-RNN.

images. Each image is annotated with forty face attributes. Different from the above two datasets, LFWA gives a fine-grained category description. We use $\{60\,\%, 10\,\%, 30\,\%\}$ random split on the whole images for training/validation/test. We use VGG-Face descriptor [18] to extract feature for each image.

4.2 Experimental Settings

Query Generation: We generate multi-attribute queries based on the dataset annotation. A query is considered to be valid when there are positive samples on train/validation and test simultaneously. We consider double and triple attribute queries for comparison. The detail information is shown in Table 1.

Table 1. Valid multi-attribute queries

Dataset	# of attributes	Double queries	Triple queries
aPascal	64	546	2224
INA	25	186	262
LFWA	40	771	9126

Evaluation Metric: We use the AUC (Area Under ROC) and AP (Average Precision) as the evaluation metric for each query. Since the number of attribute conjunctions is large, the resulting performance is hard to visualize for comparison. Therefore, we choose to use the mean AUC and mean AP to reflect the average performance of all the methods.

Comparison Methods: We compare our approach with four baseline methods: TagProp [8], RMLL [19], MARR [23] and LIFT [29]. For TagProp, we use the logistic discriminant model with distance-based weights and the number of K nearest neighbours is chosen on the validation set. For RMLL and MARR, the loss function to be optimized is Hamming loss which is also used in the original papers. Since TagProp, RMLL and LIFT do not support for multi-attribute query directly, we sum up the single attribute prediction scores as the confidence of multi-attribute query following the suggestion in [23]. The ratio parameter r of LIFT is set to be 0.1 as suggested in the paper. For our methods, the original version and the variant based on attention mechanism are presented. The optimal value of λ and the number of hidden units are chosen based on validation set by grid search.

4.3 Comparison on Image Retrieval

We calculate the mean AUC and mean AP of double and triple attribute queries for all the comparison methods. The rank for all the methods is also given for each dataset in terms of a single evaluation metric. Then we average the ranks on all the three datasets to demonstrate the overall performance. The experimental

Table 2. Experimental results (mAUC/mAP rank) for double attribute query.

Data set	Evaluate metric	TapProp	RMLL	MARR	LIFT	AC-RNN	AC-RNN-ATN
aPascal	mAUC	0.8807 5	0.8876 4	0.9040 3	0.8797 6	0.9356 2	**0.9371** 1
	mAP	0.3361 4	0.3274 6	0.3336 5	0.3383 3	0.3758 2	**0.3869** 1
INA	mAUC	0.8832 6	0.9166 3	0.8945 4	0.8902 5	0.9436 2	**0.9450** 1
	mAP	0.2269 3	0.2126 4	0.1780 6	0.1953 5	**0.2794** 1	0.2605 2
LFWA	mAUC	0.8113 6	0.8293 3	0.8210 4	0.8205 5	0.8482 2	**0.8549** 1
	mAP	0.4075 6	0.4097 5	0.4209 4	**0.4372** 1	0.4223 3	0.4370 2
Avg.Rank		5.00 6	4.17 3	4.33 5	4.17 3	2.00 2	1.33 1
Total	**AC-RNN-ATN** ≻**AC-RNN** ≻ LIFT = RMLL ≻ MARR ≻ TagProp						

Table 3. Experimental results (mAUC/mAP rank) for triple attribute query.

Data set	Evaluate metric	TapProp	RMLL	MARR	LIFT	AC-RNN	AC-RNN-ATN
aPascal	mAUC	0.8921 5	0.8988 4	0.9139 3	0.8910 6	0.9336 2	**0.9360** 1
	mAP	0.2723 3	0.2497 6	0.2582 5	0.2640 4	0.2828 2	**0.3034** 1
INA	mAUC	0.8927 6	0.9539 3	0.9375 4	0.9163 5	0.9627 2	**0.9677** 1
	mAP	0.2001 3	0.1829 4	0.1375 6	0.1521 5	**0.2743** 1	0.2726 2
LFWA	mAUC	0.8177 6	0.8367 3	0.8284 4	0.8247 5	0.8594 2	**0.8665** 1
	mAP	0.2273 5	0.2218 6	0.2355 4	0.2473 2	0.2372 3	**0.2499** 1
Avg.Rank		4.67 6	4.33 3	4.33 3	4.50 5	2.00 2	1.17 1
Total	**AC-RNN-ATN** ≻**AC-RNN** ≻ RMLL = MARR ≻ LIFT ≻ TagProp						

results are shown in Tables 2 and 3 for double and triple attribute queries respectively. Comparing the results of RMLL and MARR, we can see that MARR surpasses RMLL on different types of queries on aPascal and LFWA but fails on INA dataset. This is because the number of attribute on INA is too small and the correlation between them is not as strong as aPascal and LFWA. So the performance of MARR decreases since this method relies on strong attribute correlation. On the three datasets, we can see that our methods AC-RNN and AC-RNN-ATN achieve better performance on all types of multi-attribute queries. Therefore recurrent neural network is beneficial for modelling the complex relationship of multiple attributes. Compared to the original method, AC-RNN-ATN can leverage non-query attributes to enhance the retrieval performance and avoid the order problem by using a recurrent attention vector. It surpasses AC-RNN on both double and triple attribute queries according to the final rank and its optimal processing step is two.

Data Weighting Procedure. Images possessing all the query attributes are considered to be positive for training. Therefore, the positive samples are usually scarce when a query contains multiple attributes. To explicitly show this phenomenon, we calculate the positive sample ratio (N_m^+/N) for the queries in which the number of attributes ranges from one to three. For each type of the queries, we partition them into five parts according to the positive sample ratio and calculate the corresponding proportion for each part. The results are shown

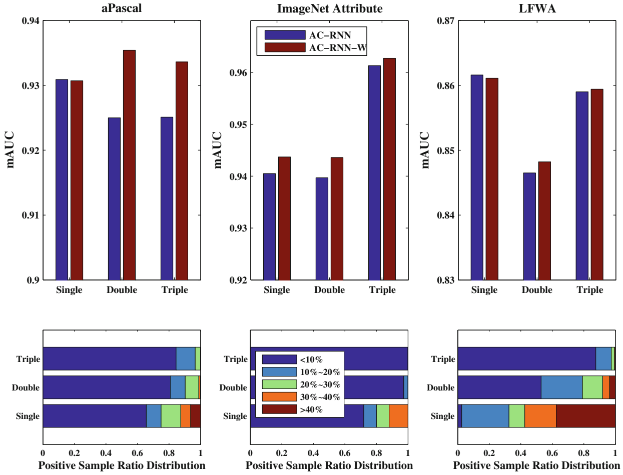


Fig. 5. Comparison on the influence of data weighting on AC-RNN. The second row shows the positive sample ratio distribution on the three datasets.

in the second row of Fig. 5. On all the three datasets, most of double and triple queries contain less than 10 % of the total data as positive samples. Therefore, how to solve data imbalance problem is essential for multi-attribute query based image retrieval. Then we train the proposed models by using the loss functions defined in Eqs. (3) and (4) respectively. The performance of the two versions with and without using data weighting for AC-RNN are shown in the first row of Fig. 5. From the results, we can see the method using data weighting procedure consistently performs better than the original version when the positive data is imbalance.

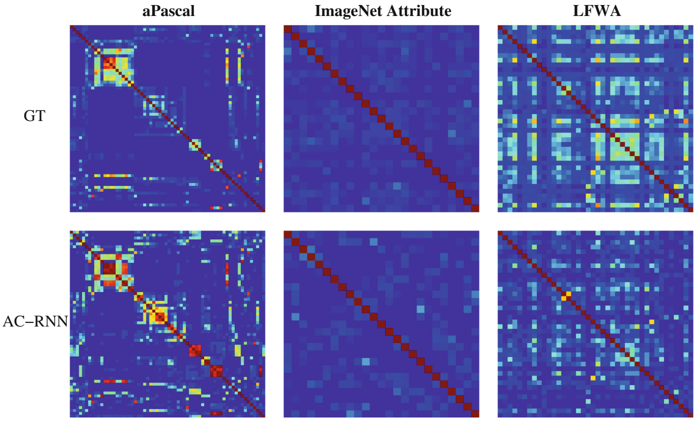


Fig. 6. Attribute similarity matrix on the embedding space.

Attribute Embedding. In this section, we validate the quality of the learned embedding matrix. We first calculate the ground truth correlation matrix which can reflect the correlation information between attributes. Let $\mathbf{R} \in \mathbb{R}^{A \times A}$ be the correlation matrix, where the correlation score between attribute i and j is computed following [24]:

$$R_{i,j} = \frac{\mathbf{Y}_{i,:}^T \mathbf{Y}_{j,:}}{\mathbf{Y}_{i,:}^T \mathbf{1} + \mathbf{Y}_{j,:}^T \mathbf{1} - \mathbf{Y}_{i,:}^T \mathbf{Y}_{j,:}}. \tag{10}$$

From the definition, we can see that two attributes are strongly correlated if they have a large number of images in common. Intuitively, the attribute embedding learned by AC-RNN is expected to reflect the correlation between attributes. So we visualize the similarity matrix of the learned attribute embeddings on all the three datasets in Fig. 6. The similarity score for a pair of attributes is calculated by using their cosine distance. Comparing the ground truth correlation matrix and the learned similarity matrix, we can see most of the correlated attributes are close on the embedding space.

4.4 Influence on Attribute Order

Attention Mechanism Based. We validate the effectiveness of learning attribute conjunction with a recurrent attention vector in this section. A promising perspective of AC-RNN-ATN is the learned conjunction is order invariant. So the dependence of attributes is no longer needed but learned in an automatic way. Moreover, the non-query attributes which are correlated with the current query are also used to generate the final representation of attribute conjunction.

The performance on the three datasets are shown in Tables 2 and 3 for double and triple attribute queries respectively. AC-RNN-ATN achieves superior performance on the both evaluation metrics. The performance gap between AC-RNN and AC-RNN-ATN is larger on aPascal and LFWA than on INA. This is because the number of attributes on INA is smaller than the other two datasets. Therefore, the correlation between query and non-query attributes is hard to exploit. Then we visualize some of the attention vectors learned by our method for both double and triple attribute queries on LFWA dataset. As shown in Fig. 7, the query contains "Chubby" will also take "Double Chin" into attention to generate the representation of attribute conjunction and an "Attractive" person with "Bushy Eyebrows" is probably "Male".

Ensemble Based. As discussion in the Sect. 3.3, attribute order for generating attribute conjunction influences the performance of AC-RNN while the optimal query order is hard to explore. Here we use an ensemble method to tackle the above problem. In detail, we repeatedly train AC-RNN for ten times on aPascal dataset. At the start of each training stage, we randomly change the attribute order in the queries instead of using the attribute order predefined by the dataset. The output matrix of each random model are combined to generate the final result for the multi-attribute query.

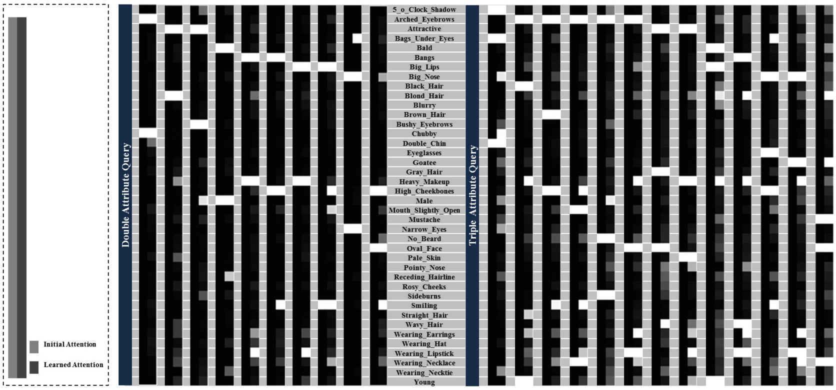
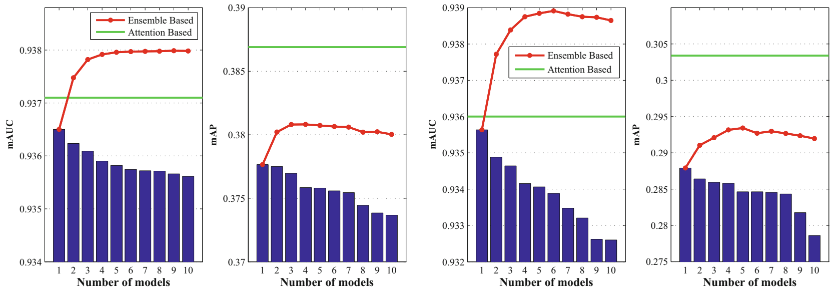


Fig. 7. Attention vector visualization on LFWA dataset.

From the results in Fig. 8, we can see the ensemble method consistently outperforms the best single model on two types of attribute queries. And the performance of ensemble based method increases rapidly at first and then becomes flat. Combining weak models may decrease the overall performance. Comparing the two methods for tackling attribute order problem, we find the ensemble based method performs better in terms of mAUC but inferior to AC-RNN-ATN according to mAP. Since our problem suffering from data imbalance, using an evaluation metric of Precision-Recall is better than Receiver-Operating-Characteristic AUC as mention in [3]. Therefore, the attention based method seems more promising.



(a) Double attribute query

(b) Triple attribute query

Fig. 8. Performance of ensemble based method in terms of mAUC and mAP on aPascal dataset. The blue bar indicates the performance of single model. (Color figure online)

5 Conclusion

We propose the attribute conjunction recurrent neural network for multi-attribute based image retrieval. Different from previous methods, our model explicitly learns the attribute embedding and generates the representation of attribute conjunction by recurrently combining the learned attribute embeddings. In addition, we propose a variant of our method using data weighting to mitigate the data imbalance problem. Finally, we have a discussion about the influence of attribute order on our method and present two methods to boost the performance based on attention mechanism and ensemble learning respectively. Experimental results on three widely used datasets show the significant improvement over the other comparison methods on all types of queries.

Acknowledgments. This work is partially supported by 973 Program under contract No. 2015CB351802, and Natural Science Foundation of China under contracts Nos. 61390515, 61272319 and 61272321.

References

1. Akata, Z., Perronnin, F., Harchaoui, Z., Schmid, C.: Label-embedding for attribute-based classification. In: 2013 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 819–826. IEEE (2013)
2. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. In: International Conference on Learning Representations (2015)
3. Davis, J., Goadrich, M.: The relationship between precision-recall and roc curves. In: Proceedings of the 23rd International Conference on Machine Learning, pp. 233–240. ACM (2006)
4. Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., Darrell, T.: Decaf: a deep convolutional activation feature for generic visual recognition. In: Proceedings of the 31st International Conference on Machine Learning, pp. 647–655 (2014)
5. Farhadi, A., Endres, I., Hoiem, D., Forsyth, D.: Describing objects by their attributes. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2009, pp. 1778–1785. IEEE (2009)
6. Frome, A., Corrado, G.S., Shlens, J., Bengio, S., Dean, J., Mikolov, T., et al.: Devise: A deep visual-semantic embedding model. In: Advances in Neural Information Processing Systems, pp. 2121–2129 (2013)
7. Graves, A.: Supervised Sequence Labelling. Springer (2012)
8. Guillaumin, M., Mensink, T., Verbeek, J., Schmid, C.: Tagprop: discriminative metric learning in nearest neighbor models for image auto-annotation. In: 2009 IEEE 12th International Conference on Computer Vision, pp. 309–316. IEEE (2009)
9. Hardoon, D.R., Szedmak, S., Shawe-Taylor, J.: Canonical correlation analysis: an overview with application to learning methods. Neural Comput. **16**(12), 2639–2664 (2004)
10. Huang, G.B., Ramesh, M., Berg, T., Learned-Miller, E.: Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical report
11. Hwang, S.J., Sigal, L.: A unified semantic embedding: relating taxonomies and attributes. In: Advances in Neural Information Processing Systems, pp. 271–279 (2014)

12. King, G., Zeng, L.: Logistic regression in rare events data. Polit. Anal. **9**(2), 137–163 (2001)
13. Kovashka, A., Parikh, D., Grauman, K.: Whittlesearch: Image search with relative attribute feedback. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2973–2980. IEEE (2012)
14. Kumar, N., Belhumeur, P., Nayar, S.: FaceTracer: a search engine for large collections of images with faces. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008. LNCS, vol. 5305, pp. 340–353. Springer, Heidelberg (2008). doi:10.1007/978-3-540-88693-8_25
15. Kumar, N., Berg, A.C., Belhumeur, P.N., Nayar, S.K.: Attribute and simile classifiers for face verification. In: 2009 IEEE 12th International Conference on Computer Vision, pp. 365–372. IEEE (2009)
16. Lampert, C.H., Nickisch, H., Harmeling, S.: Attribute-based classification for zero-shot visual object categorization. IEEE Trans. Pattern Anal. Mach. Intell. **36**(3), 453–465 (2014)
17. Liu, Z., Luo, P., Wang, X., Tang, X.: Deep learning face attributes in the wild. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 3730–3738 (2015)
18. Parkhi, O.M., Vedaldi, A., Zisserman, A.: Deep face recognition. In: British Machine Vision Conference (2015)
19. Petterson, J., Caetano, T.S.: Reverse multi-label learning. In: Advances in Neural Information Processing Systems, pp. 1912–1920 (2010)
20. Rastegari, M., Diba, A., Parikh, D., Farhadi, A.: Multi-attribute queries: to merge or not to merge? In: 2013 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3310–3317. IEEE (2013)
21. Read, J., Pfahringer, B., Holmes, G., Frank, E.: Classifier chains for multi-label classification. Mach. Learn. **85**(3), 333–359 (2011)
22. Russakovsky, O., Fei-Fei, L.: Attribute learning in large-scale datasets. In: Kutulakos, K.N. (ed.) ECCV 2010. LNCS, vol. 6553, pp. 1–14. Springer, Heidelberg (2012). doi:10.1007/978-3-642-35749-7_1
23. Siddiquie, B., Feris, R.S., Davis, L.S.: Image ranking and retrieval based on multi-attribute queries. In: 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 801–808. IEEE (2011)
24. Sigurbjörnsson, B., Van Zwol, R.: Flickr tag recommendation based on collective knowledge. In: Proceedings of the 17th International Conference on World Wide Web, pp. 327–336. ACM (2008)
25. Vinyals, O., Bengio, S., Kudlur, M.: Order matters: Sequence to sequence for sets. arXiv preprint arXiv:1511.06391 (2015)
26. Wang, Y., Mori, G.: A discriminative latent model of object classes and attributes. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6315, pp. 155–168. Springer, Heidelberg (2010). doi:10.1007/978-3-642-15555-0_12
27. Werbos, P.: Backpropagation through time: what it does and how to do it. Proc. IEEE **78**(10), 1550–1560 (1990)
28. Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., Zemel, R., Bengio, Y.: Show, attend and tell: neural image caption generation with visual attention. In: Proceedings of the 32nd International Conference on Machine Learning, pp. 2048–2057 (2015)
29. Zhang, M.L., Wu, L.: Lift: Multi-label learning with label-specific features. IEEE Trans. Pattern Anal. Mach. Intell. **37**(1), 107–120 (2015)