Improving Text-based Person Search by Spatial Matching and Adaptive Threshold

Tianlang Chen
University of Rochester
tchen45@cs.rochester.edu

Chenliang Xu
University of Rochester
chenliang.xu@rochester.edu

Jiebo Luo
University of Rochester
jluo@cs.rochester.edu

Abstract

As an important complement to person re-identification, text-based person search in large-scale database is concerned greatly for person search applications. Given language description of a person, existing frameworks search the images in the dataset that describe the same person, by computing the affinity score between the description and each image. In this paper, we first propose an efficient patch-word matching model, which can accurately capture the local matching details between image and text. In particular, it computes the affinity between an image and a word as the affinity of the best matching patch of the image toward the word. Compared with the state-of-the-art framework, it achieves competitive performance, but yields low-complexity structure. In addition, we put forward a significant limitation of affinity-based model, it is overly sensitive to the matching degree of a corresponding image-word pair. For this limitation, we feed a creative adaptive threshold mechanism into the model, it automatically learns an adaptive threshold for each word, and effectively "compress" the affinity score between a word and an image when the score exceeds the word's threshold. Extensive experiments on the benchmark dataset demonstrate the effectiveness of the proposed framework, which outperforms other approaches for text-based person search. To provide a deeper insight into the proposed model, we visualize the matching details between spatial patches of images and words of texts on typical examples, and illustrate how adaptive threshold mechanism compresses the affinity score and benefits the final rank of different images toward a text description.

1. Introduction

Text-based person search, as a new research topic of person search, has attracted remarkable attention for its strong applicability and effectiveness. As shown in Figure 1, given a query text description of a specific person, it aims to retrieve images that best match to the description from a large-scale person image database. Comparing to image-based person search (aka. person re-identification), it does not need any images of the query person, which are more difficult to obtain than text descriptions in many cases. Comparing to attribute-based person search, it can describe a person in a more accurate and comprehensive way than simply using a fixed number of predefined attributes, and it does not require the process to label the persons manually, which is tedious when the attribute list is large.

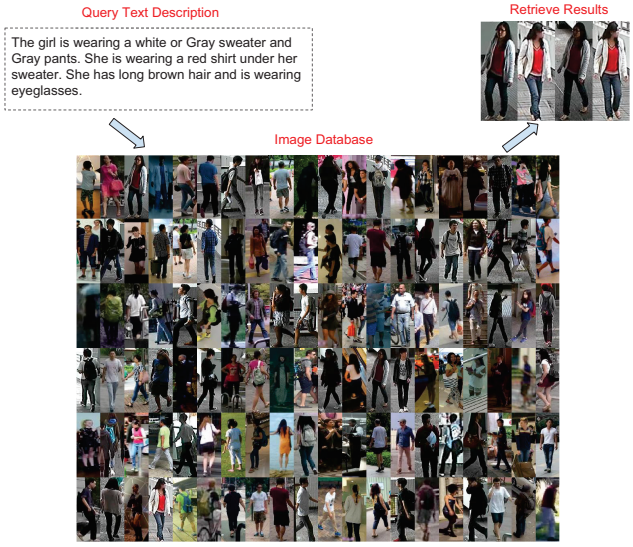

Figure 1. Overview of text-based person search.

Li et al. [6] first introduce the topic of text-based person search and propose a Gated Neural Attention mechanism (GNA-RNN) model to solve the task. Given a query text description, the GNA-RNN model computes the affinity scores between the text and each image in the database, and recommends the top few images with highest affinity scores as the search result. To obtain an image-text affinity score, the model first computes the affinity score between the image and each word in the text, then implements an attention mechanism to learn the weight of each word, and finally computes the image-text affinity score as the weighted

978-1-5386-4886-5/18/$31.00 ©2018 IEEE
DOI 10.1109/WACV.2018.00208

sum of image-word affinity scores. Their model is demonstrated to remarkably outperform other CNN-RNN models that have different network architectures or utilize different supervisions for training.

However, the GNN-RNA has two significant limitations. First, it is not sensitive to the spatial position of person's key attributes, which may lead to false high affinity for non-corresponding image-word pairs. For example, for a given word (phrase) "yellow shirt", an image containing a person with yellow shorts and a white shirt may have high affinity score, as GNA-RNN directly constructs the affinity between the word and the whole image. In this case, their model attends only to the global existence of attributes "yellow" and "shirt", even though they are not in the same region of the image or referring to the same object.

Second, the GNN-RNA is overly sensitive to the matching degree of a corresponding image-word pair. For example, the word "shirt" may gain different affinity scores with two images in which the persons wear different kinds of shirts. Because "shirt" is a key word, it has high word weight, which is learned by the attention mechanism, this will further enlarge the difference of the two images' affinity scores toward a text that contains word "shirt". Assuming that an image doesn't match one or two key words, it must not describe the same person as the text. But if it perfectly matches other key words of the text, it may still gain higher affinity score than the true corresponding image that matches all key words, but not in such perfect fashion.

In this paper, we propose a novel patch-word matching model with adaptive threshold mechanism to addresses the above limitations. First, for the patch-word matching model, instead of directly constructing the affinity between the word and global image feature, we construct the affinity between the word and local features of spatial patches from the image, and use the best matching patch-word affinity to represent the image-word affinity. Note that [5] proposes an identity-aware two-stage framework that also leverages spatial visual information on this task, but it needs two attention modules with an extra decoder LSTM sub-network to compute the image-text affinity score. We demonstrate that our proposed model can achieve competitive performance by simply incorporating a word attention module without the need of a decoder LSTM sub-network. To overcome the second limitation, we feed a novel adaptive threshold mechanism into the model. For each word, it learns a threshold that can be considered as the boundary to judge whether an image matches the word. When an image's affinity score towards a word exceeds the word's predicted threshold, we use effective approaches to reduce the gap between this affinity score and the threshold. As a result, images that match a specific word will gain similar affinity score toward this word, which decreases the model's sensitivity to the matching degree of corresponding image-word pair.

In summary, we make the following contributions:

- We propose a novel patch-word matching model for text-based person search. It starts the prediction by capturing affinity between image's local patches and text's words, and use effective approaches to figure out the image-text affinity score step by step.
- We devise an significant adaptive threshold mechanism into the framework. For each word, it predicts a threshold as a trigger to "compress" the affinity score related to the word. We test different approaches to decide when to compress and how to compress.
- We perform extensive experiments to demonstrate the effectiveness of the proposed framework, which outperforms the state-of-the-art model. We provide deep insight into patch-word matching details of several instances and illustrate how adaptive threshold mechanism makes an effect on image-text affinity score.

2. Related Work

Different from the application of person re-identification [14, 19, 3, 15, 13] and attribute-based person search [8, 11, 2], text-based person search retrieves person through natural language description. Natural language can accurately depict the person with fewer restrictions than just using predefined attributes, and it is a good complement to person re-identification when corresponding image is difficult to get. Li et al. [6] first propose the task of text-based person search, and implement a GNA-RNN model, which is demonstrated to outperform other CNN-RNN frameworks. Li et al. [5] improve the performance of search by proposing identity-aware two-stage framework, the stage-1 network screens easy incorrect matchings and also provides initial training point for training stage-2 network, and stage-2 network refines matching results by a novel latent co-attention mechanism.

For jointly textual-visual modeling on different task, incorporating the spatial information of visual modality into networks demonstrates its strong capacity to improve model's performance. For image captioning, Xu et al. [16] propose a creative framework that can decide which spatial regions of images to attend for each captioning step. For visual question answering, Yang et al. [17] present stacked attention network, it uses two attention layers to locates the relevant visual clues, and achieves better performance for question answering. For sentiment prediction, You et al. [18] feed each word as well as its corresponding image patch into a tree-structure LSTM, which remarkably improves the sentiment classification result than directly incorporating the global image feature.

Besides feeding attention mechanism that learns effective weights for different words or local image patches,

learning adaptive thresholds for different words, images or categories is also demonstrated to be useful to some tasks. For example, to improve the multi-label learning, Li et al [7] implement an independent network to learn a specific threshold for each category. When predicting the object categories for an image based on multi-label model, it includes one category if its confidence score is greater than its corresponding threshold. For image captioning, Lu et al. [9] learn a sentinel gate each step, from the combination of spatial image feature as well as the visual sentinel vector, it trades off how much new information the network is considering from the image with what it already knows in the decoder memory for this step. Motivated by their work, we learn a threshold for each word to determine whether to compress the corresponding affinity score. This mechanism significantly mitigates the oversensitivity problem for affinity-based network.

3. The Model

We introduce our text-based person search framework, where the input is an image-text pair and the output is the predicted affinity score of the pair. For an image-text pair, a higher affinity score represents higher probability that the image and text describe the same person. When we retrieve images for a text, we compute the affinity scores between the text and all the images in the candidate pool, and select images that have the highest affinity score toward the text.

3.1. Patch-word Matching Model

To capture the matching details between local regions in an image and words in a text description, we create a patch-word matching model for image-text affinity score computing. The model receives an input image-text pair and applies three steps to output an image-text affinity score. First, it computes the affinity score between each local patch of the image and each word of the text. Then, for each word, it computes its affinity score toward the image as the affinity score between the word and its corresponding best matching patch of the image. Finally, it figures out the image-text affinity score as the weighted sum of image-word affinity scores, where we implement an attention mechanism to predict the weight of each word.

An overview of our model is shown in Figure 2. Specifically, it contains four parts: an image encoder, a text encoder, a word attention sub-network and a computing part to predict the image-text affinity score. For an input image-text pair (I, T), we define $(W_1, ..., W_n)$ as the word set of the text. Same as [6], the image encoder is a VGG-16 model, which is pre-trained on person re-identification dataset. However, for image I, instead of extracting its global feature from the last fully-connected layer of the image encoder, we extract the feature of I from the last pooling layer, which is a $7 \times 7 \times 512$ tensor. In other words, the image has been down-sampled to a total of 49 patches (7×7), and each patch has a 512-dimensional feature vector. After that, we apply two m-neuron fully-connected layers to map each 512-dimensional feature into appropriate m-dimension space. We denote the jth patch's feature as $f_I^j \in \mathbb{R}^m$, where $j \in \{1, 2, ..., 49\}$. Same as [6], we set m to 512.

The text encoder is a combination of word-embedding layer and LSTM layer. For word W_i of text T, the word embedding layer first embeds it into a m-dimensional word embedding feature x_i, then LSTM layer outputs its corresponding hidden state h_i. Here, h_i can be considered as the augmented feature of word W_i with context of previous words. Similar to image encoding, we feed h_i into two m-neuron fully-connected layers that map it into appropriate space, and we denote the output as $f_W^i \in \mathbb{R}^m$.

For the jth patch and the ith word, our network computes the inner-product of f_I^j and f_W^i as the corresponding patch-word affinity score, denoted as A_{IW}^{ji}. For word W_i and image I, we denote the image-word affinity score as A_{IW}^i and compute it as the following:

$$A_{IW}^i = \max_j A_{IW}^{ji} \quad , \quad j \in \{1, 2, ..., 49\} \quad . \tag{1}$$

We use a simple example to illustrate the advantage of considering image patches and significance of Eq. 1. For a phrase "a man wears a yellow shirt", the feature of word "shirt" extracted by LSTM should keep the information of both attributes "yellow" and "shirt", because LSTM has memory of previous words. Therefore, only when both attributes exist in a single patch of an image, this patch will gain highest affinity score toward word "shirt". According to Eq. 1, our model will compute the affinity score between this image and word "shirt" as the affinity score between this patch and the word, since this patch is the best matching patch toward this word. On the other hand, if attributes "yellow" and "shirt" exist in different patches of an image, the affinity score between this image and word "shirt" should be lower, since there aren't any patches gain such high affinity score toward this word as last situation. Therefore, this design overcomes the first limitation of GNA-RNN [6], which directly extracts the image-word affinity using global image feature and thus not sensitive to the spatial position of person's key attributes.

To compute the final image-text affinity score, we incorporate a word attention sub-network into the model, which will output the weight of each word. Specifically, we obtain the attention weight of word W_i by feeding the LSTM hidden state h_i into a fully-connected layer with one output neuron. We add a sigmoid unit at the top to map the output attention weight to the range of $(0, 1)$. We use g_W^i to denote the attention weight of word W_i in the text description T. Parameters of the attention sub-network are updated with the whole model.

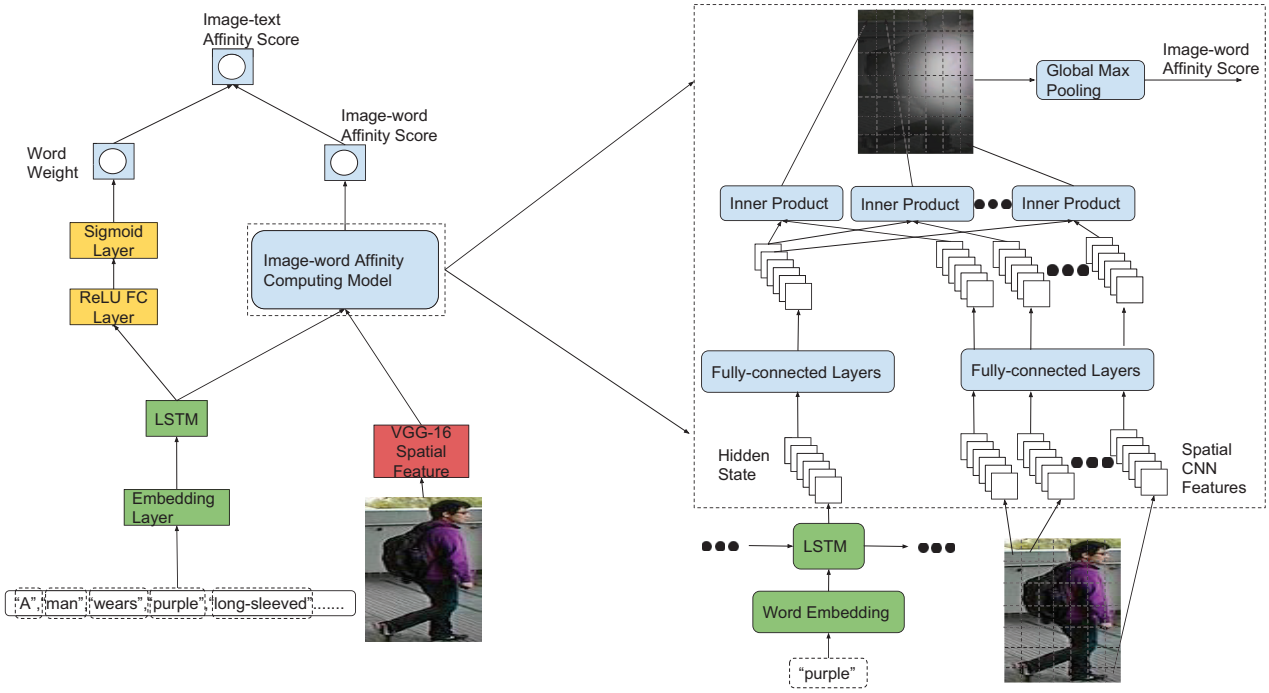

Figure 2. The structure of the proposed patch-word matching framework. It contains an image encoder (red), a text encoder (green), a word attention sub-network (orange) and a computing part to predict the image-text affinity score.

In the end, the affinity score A_{IT} for image-text pair (I, T) is computed as:

$$A_{IT} = \sigma(\sum_i (A^i_{IW} g^i_W)) \ , \tag{2}$$

where $\sigma(\cdot)$ denotes the sigmoid function mapping the image-text affinity score to the range of $(0, 1)$.

For training sample creation, we randomly choose the corresponding image-text pairs of dataset as positive samples and randomly generate non-corresponding image-text pairs as negative samples. We set the ratio between positive and negative samples to 1:3, which achieves the best performance. In the end, given the training samples, the training process minimizes the cross-entropy loss:

$$L_1 = -\frac{1}{N}\sum_{k=1}^{N}(y_k \log(A_k) + (1 - y_k)\log(1 - A_k)) \ , \tag{3}$$

where A_k denotes the predicted affinity score for the kth sample, and y_k denotes its ground truth label, with 1 representing a positive image-text pair and 0 representing an negative one. The network is trained with the Adam optimizer with a learning rate of 0.0004, and the batch size is set as 128.

3.2. Adaptive Threshold Mechanism

The final image-text affinity score is calculated as the weighted sum of image-word affinity score. If a word gains high weight, the affinity score between this word and the image will have a huge effect on the final image-text affinity score. Ideally, for two images that do match this word (e.g. word "shirt" and images with persons wearing different kinds of shirts), their affinity scores toward this word should be same, so that this word will not generate any bias for the rank relation between them. Unfortunately, their predicted affinity score toward this word may be different. This difference will be amplified by the high word weight, and thus makes bias for the final image rank. For example, if a non-corresponding image "perfectly" matches some key words of a text description, its affinity scores toward these key words will be extremely high, and this will falsely make its affinity score toward the text higher than the affinity scores of corresponding images. Motivated by this problem, we propose a novel adaptive threshold mechanism. For each word, we predict a threshold as the word's boundary to judge whether an image matches the word. If their affinity score is lower than the word's threshold, it indicates that the image does not match the word, and we do not process their affinity score. In contrast, if their affinity score is higher than the threshold, it means that the image does match the word, and we "compress" the affinity score to make it close to the threshold. As a result, for images that

match a word, their affinity scores toward the word will be close to each other, which mitigates the above problem.

Our proposed adaptive threshold mechanism consists of two steps: learning the threshold of each word, and post-processing the image-word affinity score based on the threshold. Specifically, we first train the patch-word matching model, then for each word W_{it} of the tth text T_t in the training set, we generate its threshold label as the following:

$$l_W^{it} = f(A_{I_1W}^{it}, ..., A_{I_nW}^{it}) \ , \quad I_1, ...I_n \in \mathbb{C} \ , \qquad (4)$$

where $\mathbb{C}$ is the set of all training images that describe the same person as T_t, $A_{I_mW}^{it}$ is the affinity score between image I_m and word W_{it} predicted by the patch-word matching model. We respectively set $f(\cdot)$ as the max, mean and min operation and name them as max, mean and min labeling approaches. We will compare their performance in Section 4. Words in validation/test set can be labeled in the same way as reference. On the other hand, for each word W_{it}, we can get its predicted hidden state h_{it} from the LSTM layer and its predicted attention weight g_W^{it}. Therefore, we can train a model $\mathbb{F}(\cdot)$ that minimizes the following loss function:

$$L_2 = \sum_{t=1}^{N} \sum_{i=1}^{M_t} (g_W^{it}(l_W^{it} - \mathbb{F}(h_{it})))^2 \ , \qquad (5)$$

where M_t is the total word number of text T_t and N is the total text number of training set. Here, $\mathbb{F}(\cdot)$ is a multilayer perceptron that aims to predict the threshold of a word from its corresponding predicted hidden state. For word W_{it}, we also use its predicted attention weight g_W^{it} as its learning weight to train the model, we find that feeding it into loss function leads to better result.

Therefore, for a word W_i of text T in the test set, we can predict its threshold as $\mathbb{F}(h_i)$ from its predicted hidden state h_i. After that, for image I and word W_i, we compute their compressed affinity score $\hat{A}_{IW}^i$ as:

$$\hat{A}_{IW}^i = \begin{cases} A_{IW}^i & A_{IW}^i < \mathbb{F}(h_i) \\ \mathbb{F}(h_i) + d(A_{IW}^i, \mathbb{F}(h_i)) & A_{IW}^i \geq \mathbb{F}(h_i) \, . \end{cases} \qquad (6)$$

The definition of $d(\cdot)$ decides the approach to compress the affinity score, we propose two common approaches as follows:

- We implement a hard compression by directly clipping the affinity score as the word's threshold if it exceeds the threshold:

$$d(A_{IW}^i, \mathbb{F}(h_i)) = 0 \ . \qquad (7)$$

- We implement a soft compression by using a logarithm function as following to compress the affinity score:

$$d(A_{IW}^i, \mathbb{F}(h_i)) = \frac{1}{\alpha}(\log(1 + \alpha(A_{IW}^i - \mathbb{F}(h_i)))) \ , \qquad (8)$$

where α is a hyper-parameter that is determined by the validation set, and the compression degree increases as α increases. It should have $\alpha \geq 1$ to promise that the gradient is always not larger than one, which is the prerequisite of compression.

Without doubt, for test set, if we can directly use the ground-truth threshold label of each word to compress its affinity score, the combination of min labeling and hard compression should achieve best performance, which maximizes the benefits of each corresponding image-text pair in the test set. However, in real experiments, for a word in the test set, when we use its predicted threshold to compress its affinity score toward test images, due to the error between predicted and labeled threshold, it is more possible that the model will falsely compress the image-word affinity score of corresponding image-text pairs or not compress the image-word affinity score of non-corresponding image-text pairs. The mean, max labeling and soft compression approaches reduce the probability and strength to compress affinity score of non-corresponding pairs, but also reduce the risk to compress affinity score of corresponding pairs. The best combination of approaches should stride a balance between compressing the non-corresponding image-text pairs and not compressing the corresponding image-text pairs. We will compare their performance in Section 4.

4. Experiments

We perform extensive experiments to evaluate the proposed models. We will first briefly discuss the datasets and experiment settings. Next, we compare the proposed model with state-of-the-art models on text-based person search. In the end, we provide deeper insight into the patch-word matching details and adaptive threshold mechanism.

4.1. Dataset and Experiment Settings

We evaluate our model based on CUHK-PEDES, the benchmark text-based person search dataset propose by [6]. CUHK-PEDES dataset contains 40,206 images of 13,003 person. Each image corresponds to two text descriptions. Same as [6][5], we randomly split the dataset into three subsets for training, validation, and test without having overlaps with same person. There are 11,003 persons, 33,987 images and 67,974 sentence descriptions in the training set. The validation set and test set contain 3,128 and 3,091 images, respectively, and both of them have 1,000 persons.

The top-k accuracy is adopted to evaluate the performance of person search. Specifically, given a test text description, all images in the test set are ranked according to their affinity score with the text description. If at least one of the top-k images describe the same person as the text, we define it as a successful search. Top-k accuracy represents the percentage of successful search for a specific k.

4.2. Quantitative Results

We first compare the proposed model with state-of-the-art text-based person search frameworks. [6] creates a GNA-RNN model and demonstrates that it remarkably outperforms other CNN-LSTM frameworks with different structures or utilize different supervisions for training, these frameworks include NeuralTalk [12], CNN-RNN [10], QA-Word [1], EmbBoW [20] and GMM+HGLMM [4]. [5] proposes an identity-aware two-stage framework, which utilizes a pretrain-refine strategy for training the model. It also incorporates a decoder LSTM with latent semantic attention to compute the final affinity score.

Table 1. Text-based person search results of the proposed model and other compared frameworks.

	Top-1	Top-5	Top-10
NeuralTalk [12]	13.66	–	41.72
CNN-RNN [10]	8.07	–	30.76
QAWord [1]	11.62	–	42.42
EmbBoW [20]	8.38	–	30.76
GMM+HGLMM [4]	15.03	–	42.27
GNA-RNN [6]	19.05	–	53.64
Two-stage [5]	25.94	–	60.48
PWM(ours)	25.97	48.56	60.02
PWM+ATH(max+soft) (ours)	**27.14**	**49.45**	**61.02**

"–" represents that result is not provided.

Table 1 shows the comparison between different frameworks. We first focus on the proposed patch-word matching model (PWM) without adaptive threshold mechanism. We could see that it remarkably outperforms other frameworks except the identity-aware two-stage framework. This framework also incorporates the image spatial information into the model, but using a more complex structure. It feeds spatial attention module and latent semantic module into the framework and uses a decode LSTM sub-network to figure out the affinity score. Compared with it, our patch-word matching model achieves competitive performance, but yields low-complexity structure, it just leverages a word attention module and doesn't need decoder LSTM. On the other hand, it can be seen that when we incorporates adaptive threshold mechanism into the patch-word model, the performance is further improved and outperforms all other frameworks of text-based person search for different values of k. We choose the combination of max labeling and soft compression for the adaptive threshold mechanism.

Table 2 shows the details of the proposed adaptive threshold mechanism. We compare different approaches to label the word threshold and compress the affinity score as described in Section 3.2. For threshold labeling, we respectively label a word's threshold as the maximum, mean and minimum value of the affinity scores between this word and all corresponding images. For affinity score compression, we respectively implement the hard and soft compression approaches. From Table 2, we could see that for adaptive threshold mechanism, generally, the max labeling achieves better performance than mean and min labeling, and the soft compression achieves better performance than hard compression. The combination of max labeling and soft compression achieves best performance than other approaches, it strides a balance between compressing the non-corresponding image-text pairs and not compressing the corresponding image-text pairs.

Table 2. Comparision of adaptive threshold mechanisms with different approaches to label the threshold and compress the affinity.

	Top-1	Top-5	Top-10
PWM	25.97	48.56	60.02
PWM+ATH(min+hard)	14.99	37.58	50.68
PWM+ATH(mean+hard)	22.39	46.39	58.06
PWM+ATH(max+hard)	26.10	49.30	60.62
PWM+ATH(min+soft)	25.60	48.53	60.44
PWM+ATH(mean+soft)	27.06	49.11	60.86
PWM+ATH(max+soft)	**27.14**	**49.45**	**61.02**

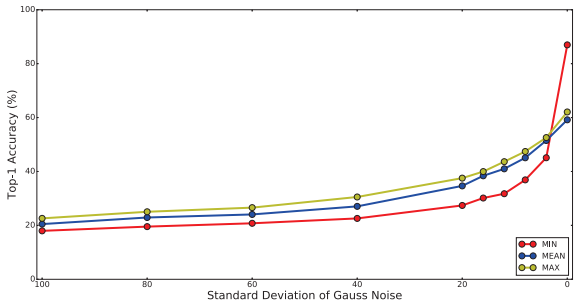

Figure 3. Top-1 accuracy comparison of different threshold labeling approaches with manually curated prediction.

To provide a deeper insight into different threshold labeling approaches, we design a manually curated task. For test image-text pairs, instead of using the predicted threshold to compress the affinity score, we manually generate the threshold of each word by adding random gaussian noise with specific standard deviation (SD) to the ground-truth threshold, and we fix the compression approach as hard compression. We show the top-1 accuracy on different situations. From Figure 3, we could see that if the SD of gaussian noise is zero, which means that we directly use the ground-truth threshold labeled by different approaches, the top-1 accuracy based on min labeling reaches nearly 87%, which is the limit of adaptive threshold mechanism. For mean and max labeling, the top-1 accuracy is about 60%. However, the accuracy based on min labeling decreases sharply as SD increases. It indicates that when there exists errors between predicted threshold and ground-truth, the error tolerance capability of min labeling is worse than mean

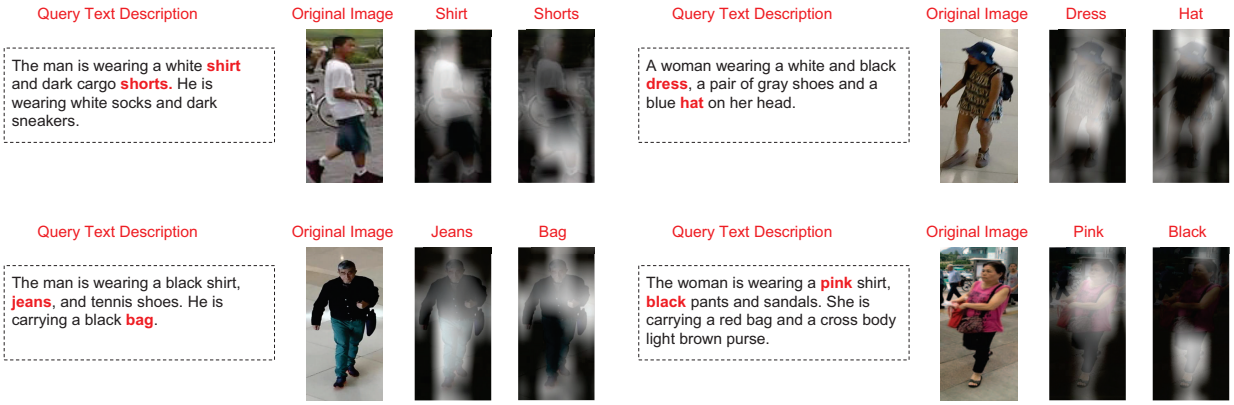

Figure 4. Affinity score maps on several typical image-word pairs. For each corresponding image-text pair, we select two words (in red) from the text and show their affinity score maps toward the image.

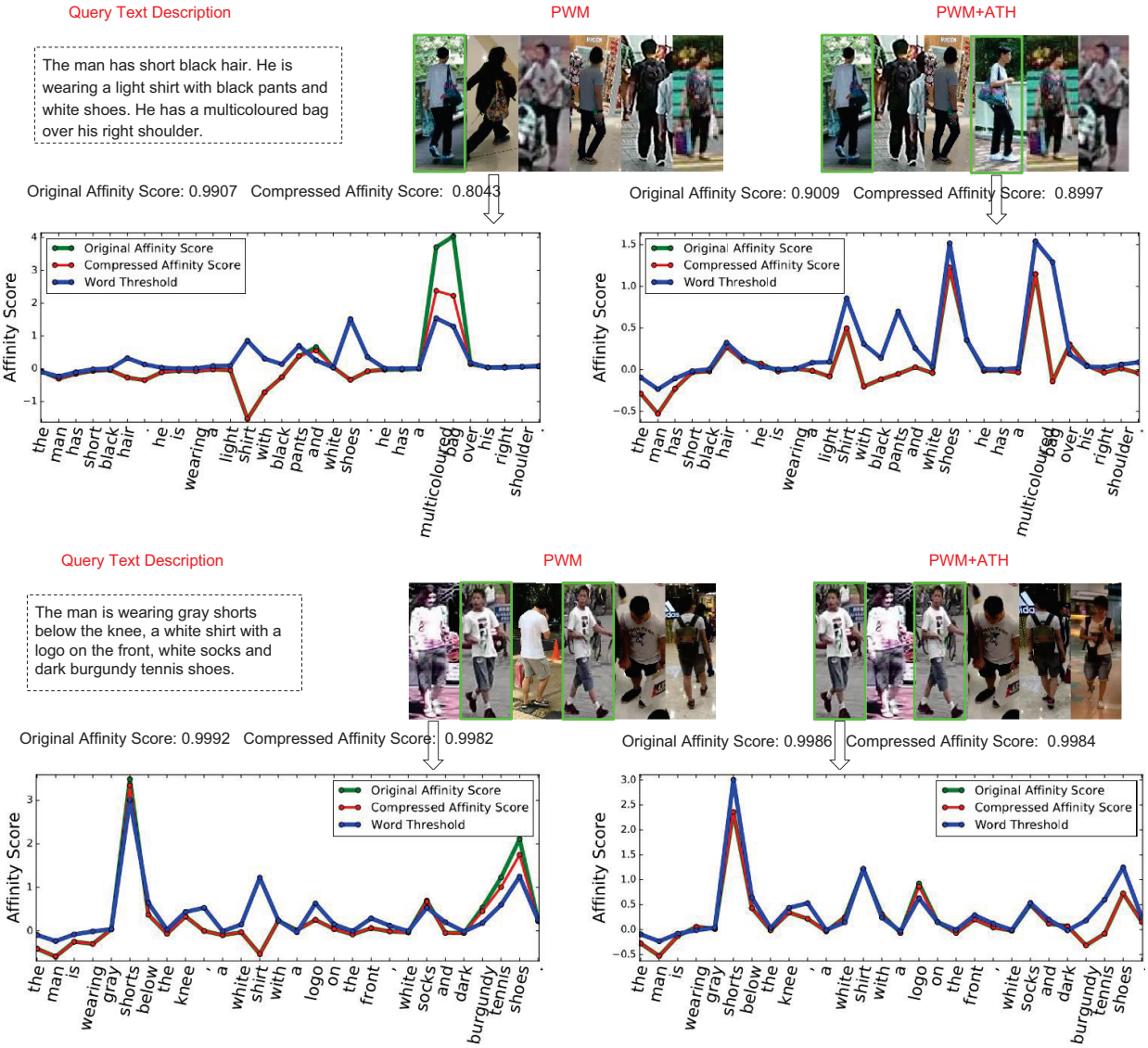

Figure 5. Visualization of weighted image-word affinity scores on several examples based on model with/without adaptive threshold mechanism. The weighted image-word affinity score is the product of image-word affinity score and corresponding word weight.

and max labeling. On the other hand, the top-1 accuracy of max labeling is better than mean labeling for different SD settings, which indicates that regardless of the threshold prediction model's performance, implementing max labeling leads to better performance. Therefore, we select max labeling approach for adaptive threshold mechanism.

4.3. Qualitative Results

For the following experiments, we provide a deeper insight into how our patch-word matching framework and adaptive threshold mechanism work. First, we visualize the matching details between patches and words based on the patch-word matching model, by showing the affinity score maps of typical image-word pairs. Specifically, for each corresponding image-text pair, we select two words from the text and visualize their affinity scores toward different patches of the image as Figure 4. From Figure 4, we could see that the proposed model can accurately capture the corresponding word and patch. For the upper left instance, toward word "shirt" and "shorts", our model respectively assigns the patches that correspond to the "shirt" and "shorts" of a person with highest affinity score. In the same way, the model can effectively capture the position of "dress", "hat", "jeans" and "bags". For the bottom right instance, in addition to nouns, we can see that it can also construct the relation between image and adjectives, such as word "pink" and "black". In fact, the great performance of patch-word matching makes the proposed computing approach for image-word affinity quite promising. Besides solving the attribute position insensitivity problem in Section 1, there is another reason for us to only select the best matching patch for image-word affinity computing. That is, for each word, the corresponding image region size is different, it will thus make bias if we compute the image-word affinity score as average/global patch-word affinity score as [6]. Further more, some other significant claims are substantiated in this experiments. For example, as an important assumption, we claim that the LSTM layer can keep the information of previous words into the current word feature. For word "bag" in the bottom left instance, its previous word is "black". From its affinity score map, we can see that it has very high affinity score toward the patch corresponds to "bag", but also has relatively high affinity score toward the patches correspond to "black shirt" and "black shoes", this demonstrates that attribute "black" is also somewhat kept in word feature of "bag".

In the end, we illustrate how adaptive threshold mechanism effectively compress the affinity score in Figure 5. For each text description, we show the top-6 images (ranked by score) with highest affinity score toward the text , the image inside a green box indicates a successful search of a corresponding image. We could see that for the instances in Figure 5, the model with adaptive threshold mechanism achieves better performance than the model without it. More importantly, for each instance, we illustrate the score compression process of two images, which are falsely ranked by the baseline model but correctly ranked by the model with adaptive threshold mechanism. Specifically, for the first instance, a non-corresponding image that describes "a woman who wears in black and has a multicolored bag" gains high affinity score toward the text for the baseline model, because her "multicolored bag" is too matched with the word "multicolored bag", and gets extremely high affinity (as the green line for words "multicolored" and "bag"). But our adaptive threshold mechanism computes a threshold for words "multicolored" and "bag" (as the blue line for words "multicolored" and "bag") and compresses the affinity score appropriately (as the red line for words "multicolored" and "bag"). This image's affinity score toward this text is thus decreased remarkably. On the other hand, for a corresponding image whose affinity score is not very high on the baseline model, the adaptive threshold mechanism almost doesn't compress its affinity score toward any words, and finally its rank is higher than the non-corresponding image. In a similar way, on the second instance, the mechanism effectively compresses the affinity score of a non-corresponding image, which is perfectly matched with words "gray shorts" and "tennis shoes". It makes this image's rank lower than a corresponding image, whose affinity score is almost not compressed.

5. Conclusions

In this paper, we focus on text-based person search, and propose a novel patch-word matching model with adaptive threshold mechanism. In particular, to compute the affinity score of an image-text pair, the patch-word matching model first captures the affinity between each local patch of the image and word of text. After that, it computes the image-word affinity score as the affinity score between the text and the best matching patch of the image. Finally, it figures out the image-text affinity score based on the word attention mechanism. This framework effectively solves the attribute position insensitivity problem with low-complexity structure. For adaptive threshold mechanism, we learn adaptive thresholds for different words as triggers to compress their affinity scores. We propose different approaches to control when to compress and how to compress. This mechanism solves the matching degree oversensitivity problem. Extensive experiments demonstrate that the proposed model outperforms the state-of-the-art approaches for text-based person search. In the end, we effectively visualize the patch-word matching details and illustrate the word-level compressing process on typical examples, which clearly shows how patch-word matching model works and how adaptive threshold mechanism improves the rank performance.

References

[1] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. Lawrence Zitnick, and D. Parikh. Vqa: Visual question answering. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2425–2433, 2015.

[2] Q. Chen, J. Huang, R. Feris, L. M. Brown, J. Dong, and S. Yan. Deep domain adaptation for describing people based on fine-grained clothing attributes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5315–5324, 2015.

[3] M. Geng, Y. Wang, T. Xiang, and Y. Tian. Deep transfer learning for person re-identification. *arXiv preprint arXiv:1611.05244*, 2016.

[4] B. Klein, G. Lev, G. Sadeh, and L. Wolf. Associating neural word embeddings with deep image representations using fisher vectors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4437–4446, 2015.

[5] S. Li, T. Xiao, H. Li, W. Yang, and X. Wang. Identity-aware textual-visual matching with latent co-attention. *Proceedings of the International Conference on Computer Vision (ICCV)*, 2017.

[6] S. Li, T. Xiao, H. Li, B. Zhou, D. Yue, and X. Wang. Person search with natural language description. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1970–1979, 2017.

[7] Y. Li, Y. Song, and J. Luo. Improving pairwise ranking for multi-label image classification. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017.

[8] Y. Lin, L. Zheng, Z. Zheng, Y. Wu, and Y. Yang. Improving person re-identification by attribute and identity learning. *arXiv preprint arXiv:1703.07220*, 2017.

[9] J. Lu, C. Xiong, D. Parikh, and R. Socher. Knowing when to look: Adaptive attention via a visual sentinel for image captioning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 375–383, 2017.

[10] S. Reed, Z. Akata, H. Lee, and B. Schiele. Learning deep representations of fine-grained visual descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 49–58, 2016.

[11] A. Schumann and R. Stiefelhagen. Person re-identification by deep learning attribute-complementary information. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2017 IEEE Conference on*, pages 1435–1443. IEEE, 2017.

[12] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3156–3164, 2015.

[13] H. Wang, S. Gong, X. Zhu, and T. Xiang. Human-in-the-loop person re-identification. In *European Conference on Computer Vision*, pages 405–422. Springer, 2016.

[14] T. Wang, S. Gong, X. Zhu, and S. Wang. Person re-identification by video ranking. In *European Conference on Computer Vision*, pages 688–703. Springer, 2014.

[15] T. Xiao, H. Li, W. Ouyang, and X. Wang. Learning deep feature representations with domain guided dropout for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1249–1258, 2016.

[16] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*, pages 2048–2057, 2015.

[17] Z. Yang, X. He, J. Gao, L. Deng, and A. Smola. Stacked attention networks for image question answering. pages 21–29, 2016.

[18] Q. You, L. Cao, H. Jin, and J. Luo. Robust visual-textual sentiment analysis: When attention meets tree-structured recursive neural networks. In *Proceedings of the 2016 ACM on Multimedia Conference*, pages 1008–1017. ACM, 2016.

[19] R. Zhao, W. Oyang, and X. Wang. Person re-identification by saliency learning. *IEEE transactions on pattern analysis and machine intelligence*, 39(2):356–370, 2017.

[20] B. Zhou, Y. Tian, S. Sukhbaatar, A. Szlam, and R. Fergus. Simple baseline for visual question answering. *arXiv preprint arXiv:1512.02167*, 2015.