LEARNING SEARCH PATH FOR REGION-LEVEL IMAGE MATCHING

Onkar Krishna Go Irie Xiaomeng Wu Takahito Kawanishi Kunio Kashino
NTT Communication Science Laboratories, NTT Corporation, Japan

ABSTRACT

Finding a region of an image which matches to a query from a large number of candidates is a fundamental problem in image processing. The exhaustive nature of the sliding window approach has encouraged works that can reduce the run time by skipping unnecessary windows or pixels that do not play a substantial role in search results. However, such a pruning-based approach still needs to evaluate the non-ignorable number of candidates, which leads to a limited efficiency improvement. We propose an approach to learn efficient search paths from data. Our model is based on a CNN-LSTM architecture which is designed to sequentially determine a prospective location to be searched next based on the history of the locations attended. We propose a reinforcement learning algorithm to train the model in an end-to-end manner, which allows to jointly learn the search paths and deep image features for matching. These properties together significantly reduce the number of windows to be evaluated and makes it robust to background clutters. Our model gives remarkable matching accuracy with the reduced number of windows and run time on MNIST and FlickrLogos-32 datasets.

Index Terms— image matching, reinforcement learning, recurrent neural network, convolutional neural network

1. INTRODUCTION

Image matching is a central research topic in image processing. Especially, finding a part of a reference image which matches to a query image is essential in a wide variety of applications such as registration [1], verification [2], tracking [3], compression [4], and stitching [5]. A desirable algorithm solving this problem should be robust enough to find the correct matches under the variations happening in the real world scenarios such as background clutter, occlusions, and geometric transformations. It should also be sufficiently fast to localize the query image from a huge number of candidate regions in a reference image within a reasonable time budget.

Exhaustive search with sliding windows gives satisfactory matching accuracy. However, it is often prohibitive, due to the huge number of windows needed to be evaluated. Most existing methods overcome this problem by introducing the idea of pruning which aims to reduce the run time by skipping unnecessary windows or pixels that will not change the final result as much as possible. For example, [6, 7, 8, 9] speedup the run time by taking into account only subset of pixels from a given query and candidate pair. [10] reduces the computation cost by skipping mismatch position based on principal orientation difference features. [11] proposes to accelerate the search process by combining random sampling of possible transformations, distance approximations, and branch-and-bound search. However, these existing methods still need to evaluate a large number of windows or pixels for identifying the target region that matches to the query, which makes the overall search process inefficient.

In this paper, we propose an image matching method that can significantly reduce the number of windows to be evaluated without loosing the matching accuracy. Our idea is to learn efficient search path from data, i.e., use machine learning to pick and evaluate only the highly prospective regions of the reference image. Overall, our method uses a recurrent neural network model to sequentially search promising regions over the reference image that are expected to match to the query. The network is trained through reinforcement learning, and thanks to its nature our model can be trained in an exploratory manner; explicit supervised information such as the location of the target regions and class labels is not necessary. Furthermore, our model jointly learns image features in an end-to-end manner, which are useful for measuring similarities. It achieves accurate matching by validating only an extremely small number of windows. We evaluated our model on MNIST and FlickrLogos-32 datasets and show that it gives remarkable matching accuracy with reduced number of windows and run time.

2. METHOD

Suppose we are given a pair of a query and a reference images denoted by $\mathcal{Q}$ and $\mathcal{R}$, respectively. Our task is to localize the "target region" represented by $\mathcal{Q}$ which exists at a certain position l_g on $\mathcal{R}$. We denote by $\mathcal{R}(l)$ the region of $\mathcal{R}$ at the position l. We approach to this task by sequential search. More specifically, our goal is to determine the sequence of search window locations $\{l_t\}_{t=0}^{T}$ so that it can correctly localize the position of the target region l_g with small T, where T [1] is the length of the sequence.

We propose a machine learning approach using a recurrent neural network. The schematic overview of our model is

[1]For simplicity, we assume that the window size is fixed throughout the paper. Extension to the case of arbitrary sized window is straightforward.

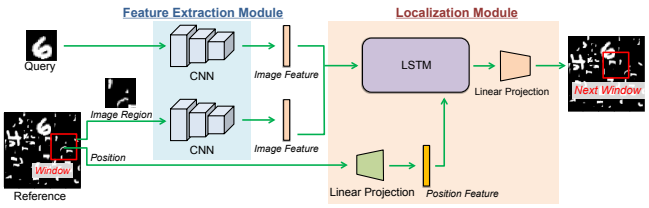

Fig. 1. Overview of our proposed model. Our model has two major modules: feature extraction module and localization module.

illustrated as in Fig. 1. Our model consists of two major modules which we call *feature extraction module* and *localization module*, respectively. The basic behavior of our model at each time step t can be summarized as follows. The feature extraction module first extracts the image features of $\mathcal{Q}$ and $\mathcal{R}(l_t)$, denoted by $f(\mathcal{Q})$ and $f(\mathcal{R}(l_t))$, respectively. The similarity between $\mathcal{Q}$ and $\mathcal{R}(l_t)$ is measured by using these features for matching. Then the two image features $f(\mathcal{Q})$ and $f(\mathcal{R}(l_t))$ as well as the current window location l_t are fed to the localization module that estimates the next position of the window, l_{t+1}. These two sub-processes are repeated sequentially until the maximum number of iterations T is reached. We detail the two modules in Sec. 2.1.

Training of our model is tricky, because the localization is performed in a sequential manner. The current decision at a time is made depending on all the past decisions, so the quality of the decision at each time step cannot be evaluated independently from the others. Therefore, typical supervised or unsupervised learning methods which assume the samples to be i.i.d. cannot be applied to our case. Fortunately, this process can be modeled as a partially observable Markov decision process (POMDP), so the training can be performed in a reinforcement learning manner. In this paper, we propose a method that can jointly learn image features as well as search paths, i.e., sequence of the window locations to be searched, in a unified framework. Our algorithm is inspired by the recurrent attention models [12, 13, 14] which are proposed for image recognition. Unlike these, our model is customized to image matching task and designed to learn image features effectively for similarity matching. Furthermore, the entire model can be trained with less supervision, i.e., unlike these models, ours does not require any class labels for training. We give the detail of our algorithm in Sec. 2.2

2.1. Details of Modules

As shown in Fig. 1, our model is basically a CNN-LSTM model and uses the CNN for *feature extraction* and LSTM for *localization*. We hereafter give the details of these two major modules one-by-one.

Feature Extraction Module. This module extracts the image features from $\mathcal{Q}$ or $\mathcal{R}(l_t)$. It consists of two identical CNNs with the same parameters; one for $\mathcal{Q}$ and the other for $\mathcal{R}(l_t)$. The CNN is designed to have a sequence of five Conv-ReLU layers (ReLU activation after 2D convolutions) followed by a global average pooling. This is fully-convolutional so does not have any fully-connected layers. The advantage of this configuration is that it can extract a feature vector of the same length from an arbitrary size of input image.

Localization Module. The localization module is the most salient feature of our model. The main component of the module is LSTM that sequentially predicts the next location l_{t+1} based on three external inputs including the two image features $f(\mathcal{Q})$ and $f(\mathcal{R}(l_t))$ and the current window location l_t. In our implementation, the three inputs are concatenated to form a single vector and then fed to the LSTM. However, since the location l_t is lower-dimensional (2D) compared to the image features (128 or more), the resulting vector is dominated by the elements of the image features, which makes it difficult to capture useful positional information. To avoid this, by a linear projection, l_t is first encoded to a *position vector* which has the same dimensions as the image feature and then concatenated with them. The resulting hidden state h_t of the LSTM is translated to the expected location of the next window $\hat{l}_{t+1}$ by another linear projection. We assume that the actual position of the next window l_{t+1} is a stochastic variable that follows a Gaussian distribution, where $\hat{l}_{t+1}$ gives the mean vector. Specifically, l_{t+1} is obtained as a sample from the distribution $\mathcal{N}(\hat{l}_{t+1}, \lambda I)$ as $l_{t+1} \sim \mathcal{N}(\hat{l}_{t+1}, \lambda I)$, where I is the identity matrix and λ is a hyperparmeter.

The initial position of the window l_0 is determined in a similar way used in [13, 14]. Specifically, we first extract the image features of the (down-sampled) global reference image, $f(\mathcal{R})$, by using the feature extraction module and then feed it to another linear projection which is analogous to the context network used in [13, 14] to generate l_0. Both of the $f(\mathcal{R})$ and l_0 are fed to the LSTM to predict the next position l_1.

2.2. Model Training

Let $\Theta = \{\theta_f, \theta_l\}$ be the parameters of the whole model, where θ_f and θ_l are the parameters of the feature extraction module and the localization module, respectively. We use reinforcement learning to tune Θ. Since the location l_t is determined sequentially in our method, l_t is determined with the condition on all the past locations visited. For notational simplicity, we use $s_{t-1} = \{\{l_\tau\}_{\tau=1}^{t-1}, \mathcal{Q}, \mathcal{R}\}$. The *policy* of our model then can be represented as a conditional distribution $\pi(l_t|s_{t-1}; \Theta)$. Now our goal is to maximize the total reward $R = \sum_{t=1}^{T} r_t$ w.r.t. Θ. A natural choice for the reward function r_t in our case would be the success or failure of the search at time t: $r_t = 1$ if and only if the window at time t correctly captures l_g, and $r_t = 0$ otherwise. The expected value of the total reward is given as

$$J(\Theta) = \mathbb{E}_{p(s_T;\Theta)}[R], \tag{1}$$

where $p(s_T; \Theta)$ is the probabilistic distribution of s_T which depends on the policy. Although the gradient w.r.t. Θ is non-

trivial, it can be approximately computed by sampling the sequences of $\{l_t, s_{t-1}\}_{t=1}^{T}$ from the policy in a similar way to Monte-Carlo approximation, which gives

$$\nabla_\Theta J(\Theta) \approx \frac{1}{M}\sum_{i=1}^{M}\sum_{t=1}^{T}\nabla_\Theta \log \pi(l_t^i|s_{t-1}^i;\Theta)R^i. \quad (2)$$

where M is the number of sample sequences. By using this, Θ can be iteratively updated through gradient ascent.

Training our model only with reinforcement learning often makes the prediction results unstable. Hence, we involve another loss function to improve the stability of the learning process. Specifically, we require the image features extracted by the feature extraction module to correctly measure the similarity between $f(\mathcal{Q})$ and $f(\mathcal{R}(l_t))$, i.e., the distance between the two features should be small for matching pairs and large for non-matching pairs. To this end, we impose a contrastive loss function on the feature extraction module as done in widely-used Siamese networks.

$$L(\theta_f) = \sum_{t=1}^{T} r_t d^2 + (1-r_t)\max\{0, m-d\}^2, \quad (3)$$

where $d = ||f(\mathcal{Q}) - f(\mathcal{R}(l_t))||$ and m is a margin. This is piecewise differentiable w.r.t. θ_f and so can be easily optimized with gradient descent.

3. EXPERIMENTS

3.1. Datasets

We use two benchmark datasets, MNIST[2] and FlickrLogos-32[3] [15], in our experiments for evaluation.

MNIST. We considered three experimental protocols on the basis of the MNIST dataset. The first protocol is referred to as **Translated MNIST**, in which each reference image was generated by placing the query, i.e., a 28×28 digit, in a random location of a 100×100 blank patch. The second protocol is called **Cluttered MNIST** and is used to evaluate the robustness of the methods as regards background clutter. Here, each reference image was generated by adding random 9×9 subpatches from other random digits to random locations of a Translated MNIST reference image. Finally, we placed a randomly chosen 28×28 digit, which is different from the targett digit, at a random location of each cluttered MNIST reference image, leading to our third protocol, **Mixed MNIST**.

All query-reference pairs were prepared for all the three MNIST variants by considering each 100×100 digit as reference and then selecting a 28×28 query of the same digit from a master set of 10 centered clean numbers from 0 to 9. Following the standard split of MNIST, we used $10,000$ query and reference pairs for testing and the remaining $60,000$ pairs for training.

FlickrLogos-32. This dataset consists of $2,240$ images of 32 different logos with 70 images per logo. In our experiments, the training and the test sets are composed of $2,000$ and 240 query-reference pairs, respectively. Each pair was generated by considering a logo image in the dataset as the reference and a tightly cropped logo of the same brand as the query. In total, we have 32 query images, each corresponding an individual logo. All the reference images were resized to half of their original size; each query was then resized to the same size of the logo in the reference image.

3.2. Experimental Setup

Performance Metrics. We evaluated our approach in terms of accuracy and speed. Given a query and a reference image, our model outputs a predicted window corresponding to a region in the reference image, which is used to evaluate the accuracy. In particular, an image matching is considered as being successful if the intersection over union (IoU) between the predicted window and the ground-truth window is greater than 0.5, following the same standard as in the object detection literature. We report the success rate which represents the ratio of the number of image pairs with correct matches to all the pairs. The efficiency is evaluated in terms of two measures. One is the number of windows evaluated, and the other is run time required for processing each query-reference pair.

Baselines. We compared our method with two existing image matching methods, namely BBS [7] and MTM [16]. We use the codes provided by the author groups. We make sure that the same hardware environment is used for each method and that hyper-parameters of these methods are carefully tuned.

Learning Configurations. We trained the proposed model from scratch using Adam with a batch size of 64 for MNIST and 1 for FlickrLogos-32. The learning rate was kept in the range $[10^{-4}, 10^{-3}]$ with an exponential decay. The variance hyperparameter of the Gaussian λ, which is used for sampling out the next location, is fixed to 0.22.

3.3. Results

For all the datasets, the success rate, the number of windows evaluated, and the run time are reported in Tables 1, 2 and 3, respectively.

Results on Translated MNIST. First, as can be seen in the Table 1, the success rate of our method is best among all the baselines. The maximum gain of our method over the baselines reaches 0.25 on BBS and 0.27 on MTM. The results clearly demonstrate that our model is able to learn search path very accurately on Translated MNIST dataset. Second, as shown in Table 2, our model clearly outperforms the baseline

[2]http://yann.lecun.com/exdb/mnist/

[3]http://www.multimedia-computing.de/flickrlogos/

Table 1. Image matching success rate.

Dataset	Translated MNIST	Cluttered MNIST	Mixed MNIST	Flickr Logos-32
Ours	**0.95**	**0.91**	**0.88**	**0.39**
MTM [16]	0.68	0.20	0.15	0.28
BBS [7]	0.70	0.11	0.08	0.36

Table 2. Number of windows evaluated to localize a query.

Dataset	Translated MNIST	Cluttered MNIST	Mixed MNIST	Flickr Logos-32
Ours	**6**	**8**	**8**	**6**
MTM [16]	5329	5329	5329	18298
BBS [7]	10000	10000	10000	40112

methods in terms of the total number of candidate windows evaluated to localize the query. Our method just evaluate 6 candidate windows, whereas the baseline methods evaluates in thousands. The advantage of processing only a few window reflects in the run time. Although the run time is not directly proportional to the number of windows processed since each methods have different computation requirement for every pixel, ours is competitive to or much faster than the other two methods, while yielding much better matching accuracy.

Results on Cluttered and Mixed MNIST. In terms of success rate in matching, our method is the best among all the baselines, as shown in Table 1. This suggests that our model can successfully learn search path even in the presence of a wide range of background clutters. BBS performs the worst among the three methods. This is because the matching between query and candidate windows in BBS is evaluated according to the consistency of the distributions of pixels; two windows were determined as a matching pair if their pixel distributions in (x, y, R, G, B) space are similar. This strategy is not effective on Cluttered and Mixed MNIST where the noise may have the same underlying distribution as the target. Our method can accurately localize the query object in just 8 candidate windows for both Cluttered and Mixed MNIST. Also, our method is competitive or superior in run time.

Results on FlickrLogos-32. This dataset is the most challenging because the targets are different from the queries due to a wide variety of scale changes, viewpoint changes, and deformations. Table 1 shows that our method outperforms all the baselines in accuracy. In terms of run time, the gain of our method is significant compared to the cases of MNIST datsets. This is because the sizes of the reference images are larger than those of MNIST and the run time of BBS and MTM is almost linear in the reference image size. The run time of our method only depends on the number of windows to be evaluated, which is far smaller than the two baselines. This suggests that our approach is more efficient when it is applied to more realistic and larger size images.

Qualitative Results. Figure 2 shows some qualitative results. It demonstrates the excellent ability of our model in learning

Table 3. Average run time in milliseconds.

Dataset	Translated MNIST	Cluttered MNIST	Mixed MNIST	Flickr Logos-32
Ours	**1**	**3**	**4**	**6**
MTM [16]	2	**3**	5	230
BBS [7]	132	141	148	2390

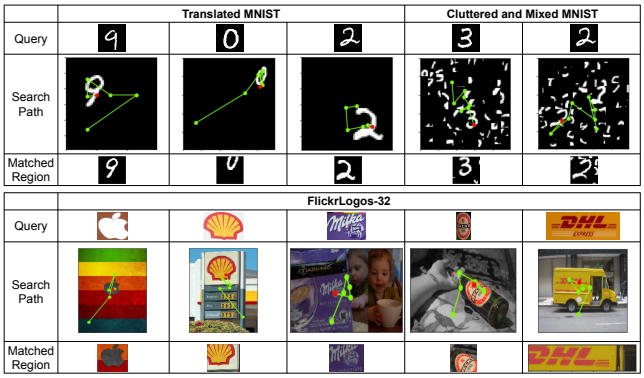

Fig. 2. Qualitative results. For a given query-reference pair, the example shows the search path traced by our model in order to localize the query. The search path is overlaid in green lines. The red dot indicates the predicted location at the last time step. The window cropped from the last attended location is the search output of our model.

the search path. The results on MNIST datasets show that despite the increased level of search difficulty due to clutter, our model still can successfully localize the query object with almost the same number of candidate windows evaluated. Similarly, from the FlickrLogos-32 examples, it can be seen that our method can successfully find the target logos even if they are heavily different in their colors and poses. For instance, in the case of the left most example (*Apple* logo), our model successfully localizes the target despite the different appearance between query and reference images. This can be possible because our model jointly learns search paths and effective deep features for matching.

4. CONCLUSIONS

In this paper, we attempted to address the fundamental problem of matching a query to a region in a reference image by introducing a novel end-to-end learning based method. The proposed method was based on CNN-LSTM that sequentially outputs the next location towards the target region in each iteration. The model has several appealing properties, due to its ability of joint feature and search path learning. First, the number of candidate windows processed to localize the query is far smaller than existing methods, which leads to faster image matching especially for large images. Second, as can be seen in the experimental results, our model is able to localize the query even in severely cluttered reference images.

5. REFERENCES

[1] Arthur Ardeshir Goshtasby, *2-D and 3-D image registration: for medical, remote sensing, and industrial applications*, John Wiley & Sons, 2005.

[2] Shayan Modiri Assari, Haroon Idrees, and Mubarak Shah, "Human re-identification in crowd videos using personal, social and environmental constraints," in *European Conference on Computer Vision*. Springer, 2016, pp. 119–136.

[3] Zhibin Hong, Zhe Chen, Chaohui Wang, Xue Mei, Danil Prokhorov, and Dacheng Tao, "Multi-store tracker (muster): A cognitive psychology inspired approach to object tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 749–758.

[4] Stuart Inglis and Ian H Witten, "Compression-based template matching," in *Data Compression Conference, 1994. DCC'94. Proceedings*. IEEE, 1994, pp. 106–115.

[5] Jing Zhang, Guangxue Chen, and Zhaoyang Jia, "An image stitching algorithm based on histogram matching and sift algorithm," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 31, no. 04, pp. 1754006, 2017.

[6] Vasudevan V Vinod and Hiroshi Murase, "Focused color intersection with efficient searching for object extraction," *Pattern recognition*, vol. 30, no. 10, pp. 1787–1797, 1997.

[7] Tali Dekel, Shaul Oron, Michael Rubinstein, Shai Avidan, and William T. Freeman, "Best-buddies similarity for robust template matching," in *Proc. CVPR*, 2015.

[8] Itamar Talmi, Roey Mechrez, and Lihi Zelnik-Manor, "Template matching with deformable diversity similarity," in *Proc. CVPR*, 2017.

[9] Ofir Pele and Michael Werman, "Accelerating pattern matching or how much can you slide?," in *Proc. ACCV*, 2007.

[10] Jichao Jiao, Xin Wang, Zhongliang Deng, Jichang Cao, and Weihua Tang, "A fast template matching algorithm based on principal orientation difference," *International Journal of Advanced Robotic Systems*, vol. 15, no. 3, 2018.

[11] Simon Korman, Daniel Reichman, Gilad Tsur, and Shai Avidan, "Fast-match: Fast affine template matching," in *Proc. CVPR*, 2013.

[12] Volodymyr Mnih, Nicolas Heess, Alex Graves, and Koray Kavukcuoglu, "Recurrent models of visual attention," in *NIPS*, 2014.

[13] Jimmy Ba, Volodymyr Mnih, and Koray Kavukcuoglu, "Multiple object recognition with visual attention," in *Proc. ICLR*, 2015.

[14] Artsiom Ablavatski, Shijian Lu, and Jianfei Cai, "Enriched deep recurrent visual attention model for multiple object recognition," in *Proc. WACV*, 2017.

[15] Stefan Romberg, Lluis Garcia Pueyo, Rainer Lienhart, and Roelof Van Zwol, "Scalable logo recognition in real-world images," in *Proc. ICMR*, 2011.

[16] Yacov Hel-Or, Hagit Hel-Or, and Eyal David, "Fast template matching in non-linear tone-mapped images," in *Proc. ICCV*, 2011.