OS1A-3

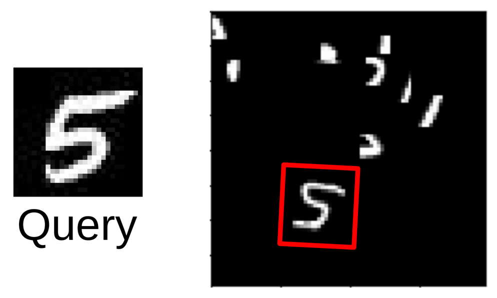
Deep Reinforcement Template Matching

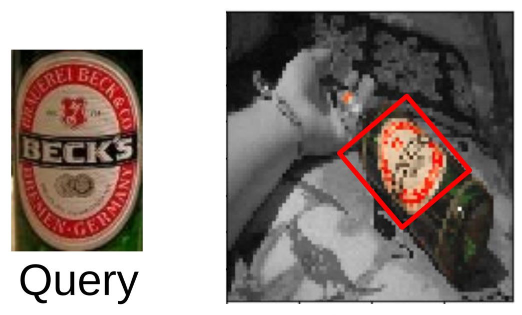
Onkar Krishna Go Irie Xiaomeng Wu Takahito Kawanishi Kunio Kashino

NTT Corporation

Overview

- **Template Matching**: Find a part of a reference image that matches to a query image.





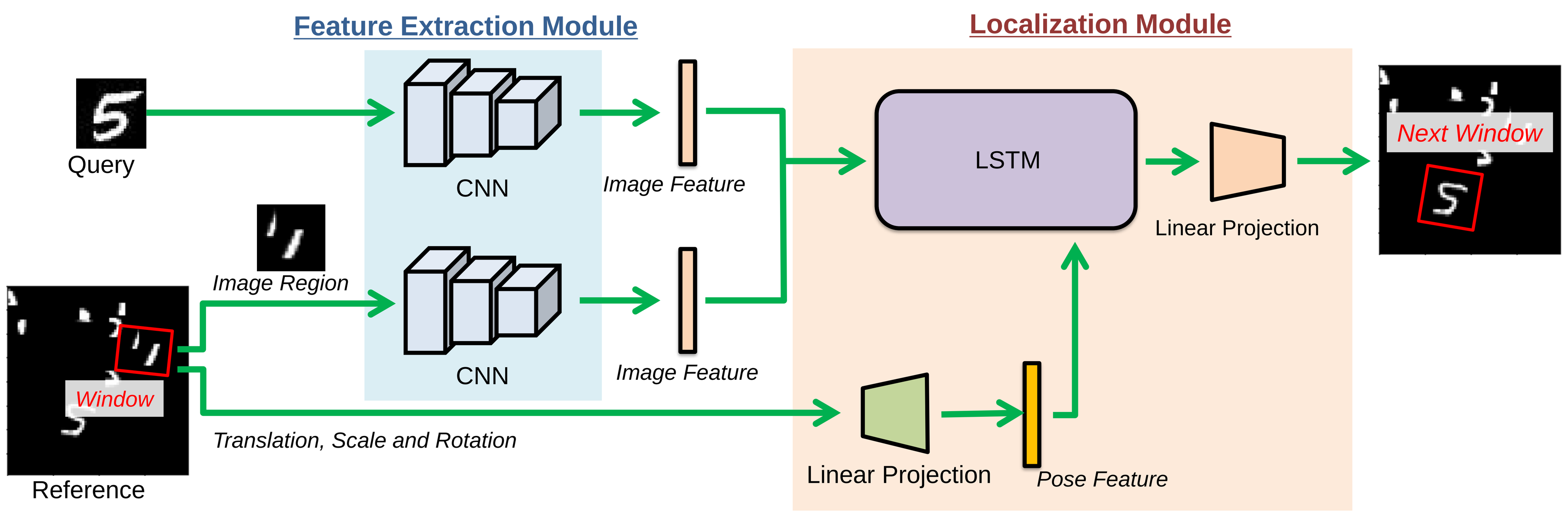
A desired algorithm should be ***fast*** and ***robust*** to noise, e.g., background, illumination change, and geometric transformation.

- We propose a **deep reinforcement learning approach**
  - **Joint learning of image features and search path**: Pick and evaluate only the highly prospective regions of the reference image in a sequential manner.

✔ **Good balance between speed and accuracy**
✔ **Robust to background clutters and geometric transformations**
✔ **Do not requires any class label or exact pose supervision**

Model Architecture

Our model has **Feature Extraction Module** and **Localization Module**



Feature extraction module

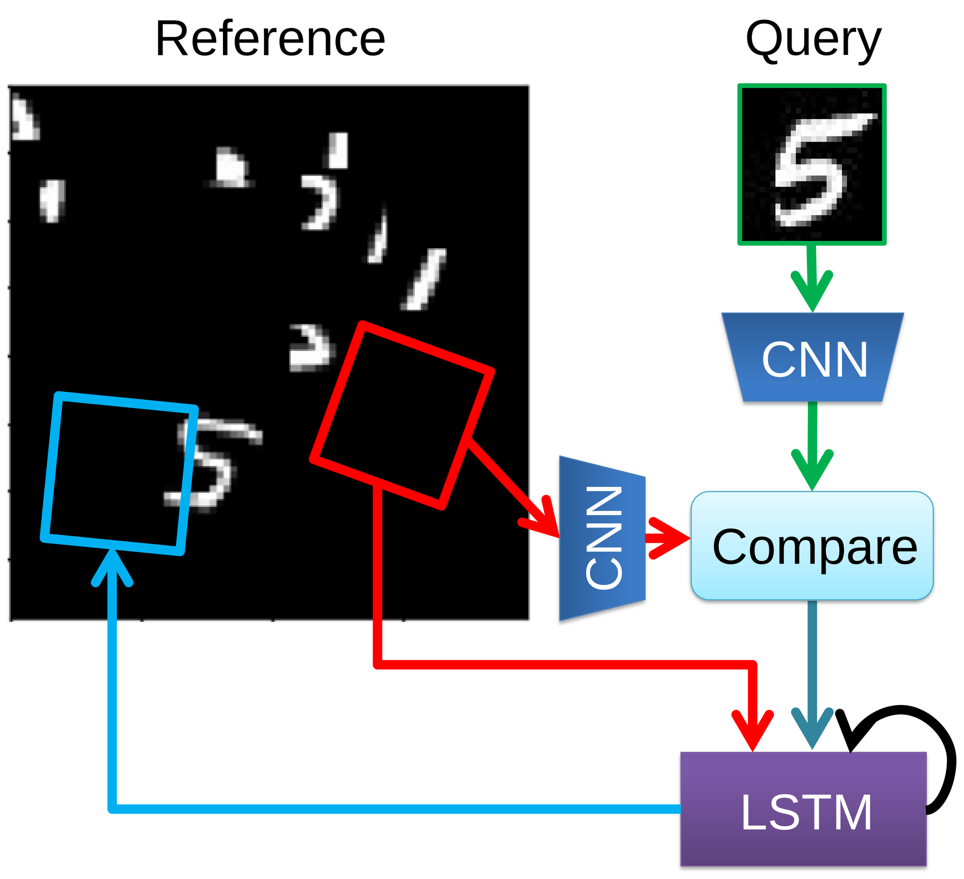
- Extracts the image features from query and reference image region.
- Consists of two identical CNNs with the same parameters which have a sequence of five Conv-ReLU layers followed by a global average pooling.

Localization module

- Has an LSTM that sequentially predicts the next window pose based on three external inputs including two image features and current window pose.

This design allows us to jointly learn the search path and effective deep features for matching!

Algorithm Behavior



Step 0
Choose the initial window pose

Step 1
Compare the features between the chosen window and the query

Step 2
Choose the next window pose given the features and the current window pose

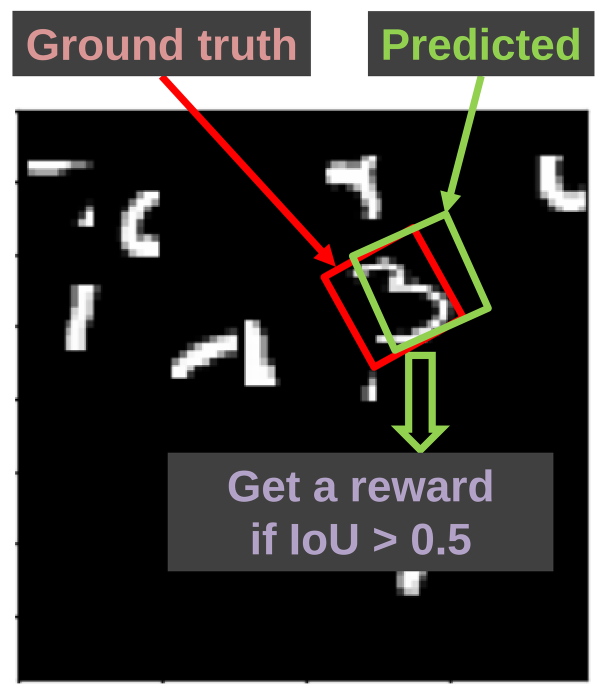
Step 3
Repeat until the number of trials reaches the limit

Learning Strategy

Combination of **reward maximization** and **feature loss minimization**

a. Reward maximization

- Get a reward "1" if IoU > 0.5, otherwise "0"



- Maximize the expected reward based on the policy gradient

b. Feature loss minimization

- Contrastive loss to learn good features for matching

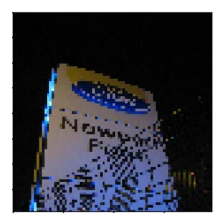
$$L = \begin{cases} d^2(q,g) & \text{If "Success"} \\ \max\{0, m - d(q,g)\}^2 & \text{otherwise} \end{cases}$$

$d(q,g)$: Euclidian distance between query q and reference window g

Experiments

Datasets

- We use three datasets to evaluate our method.

	Transformed MNIST	Transformed+Cluttered MNIST		FlickrLogos-32	
Query					
Reference					
# data	60K training 10K testing	60K training 10K testing		2K training 240 testing	

Quantitative Results

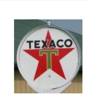
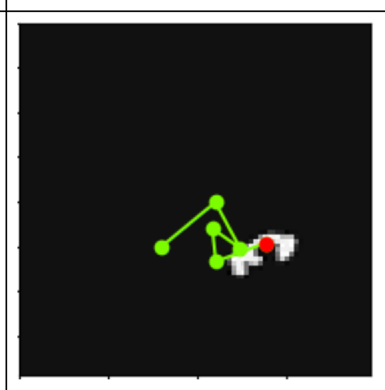
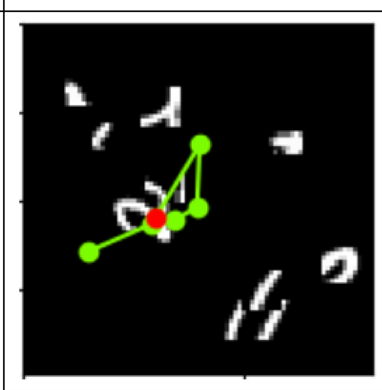
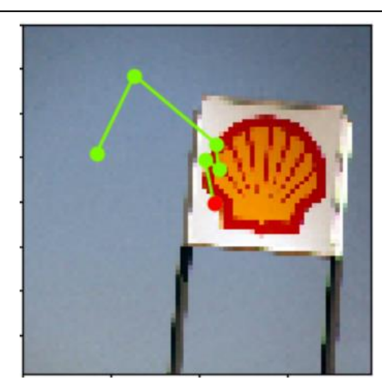
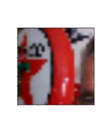
Success rate* (run time in milliseconds)

Dataset	Transformed MNIST	Transformed+ Cluttered MNIST	FlickrLogos-32
Ours	**0.89** (3.8)	**0.85** (4.0)	**0.34** (26.2)
[Yacov+, ICCV11]	0.51 **(1.1)**	0.18 **(1.0)**	0.10 **(5.2)**
[Tali+, CVPR15]	0.56 (90.1)	0.20 (90.3)	0.31 (110.6)

*Search is judged as successful if IoU > 0.5

✔ **Ours is robust to background clutter and able to handle geometric transformations.**
✔ **While [Yacov+, ICCV11] is slightly faster, ours is much more accurate with a slight expense of run time.**

Qualitative Results

Query				
Search Path				
Found Region				

✔ **The window converges to the target region once a part of target region is captured, otherwise it randomly moves to next location.**

Conclusions

We proposed a reinforcement learning approach to template matching.

- **Accuracy/speed**: Our method achieved better matching accuracy with highly competitive search speed.
- **Explorative learning**: Our model jointly learns search path and good image features for matching in a reinforcement learning manner.