A spatiotemporal model with visual attention for video classification

Mo Shan and Nikolay Atanasov
Department of Electrical and Computer Engineering
UC San Diego

Motivation

Classification tasks often involve rotation and scale changes:

1. Self-driving cars need to classify scenes that contains small and large vehicles for loop closure
2. Robots have to infer the classes of objects rotated in different ways in order to grasp them

Problem Formulation

Given a training dataset that contains videos $D_{train} = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$, where $x_i \in \mathbb{R}_{>0}^{w \times h \times c \times t}$ is the i_{th} video, w, h, c is the size for each frame, t is the number of frames, and $y_i \in \{1, 2, ..., K\}$ is the i_{th} class label, we aim to train a classifier $\mathbf{C}$ for unseen data $(x_i, y_i) \in D_{test}$, such that $\mathbf{C}(x_i) = y_i$.

Proposed model

Architecture

Motivated by the advantages of both types of networks, this paper proposes a spatiotemporal model in which CNN and RNN are concatenated, as shown in Fig. 1.

1. CNNs are used to extract hierarchical features from frames
2. Attention modules could handle geometric transformations
3. RNNs are good at harnessing the temporal dependency in sequential visual input

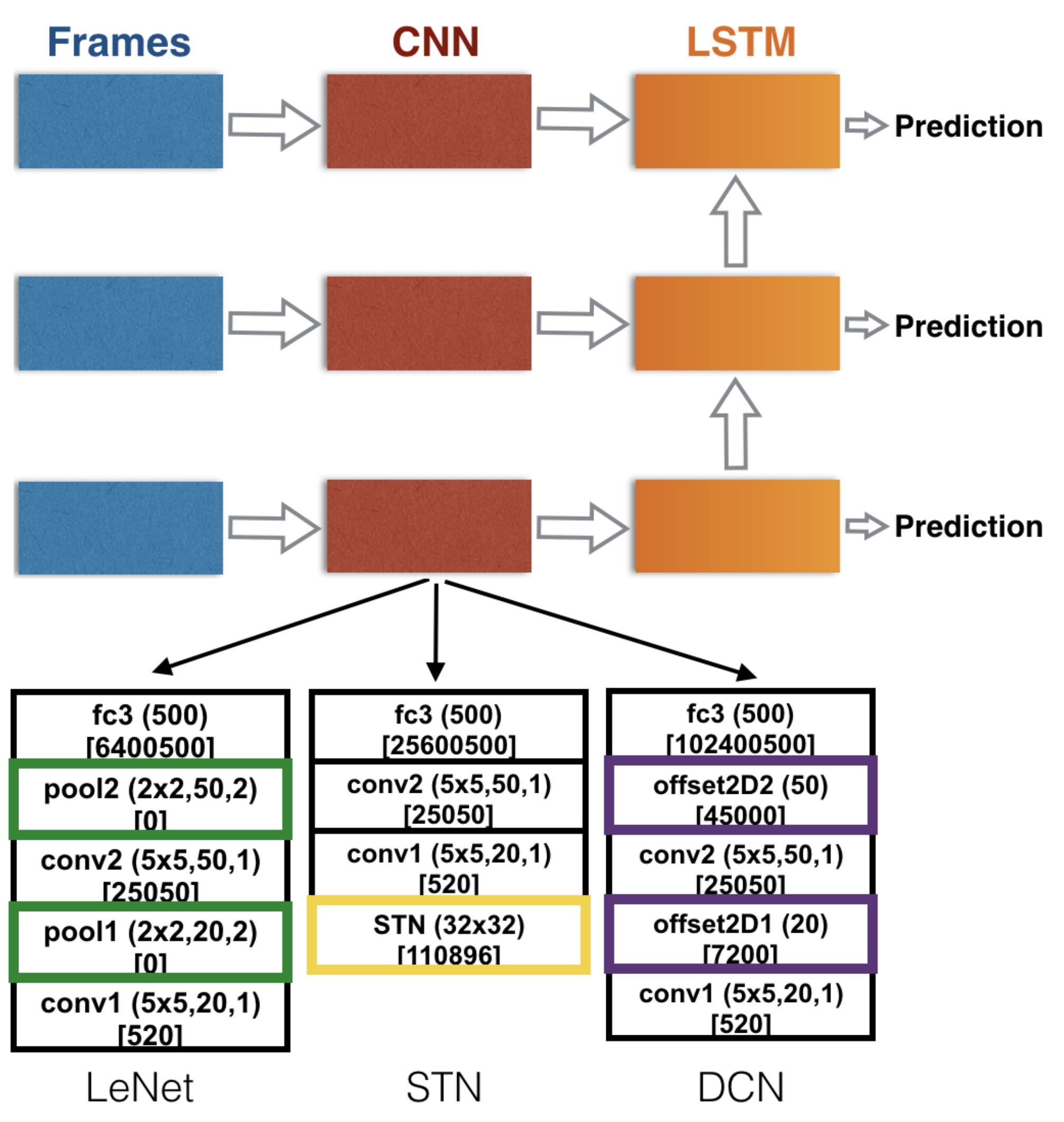

Figure 1: Architecture of the proposed model: The spatiotemporal model contains a CNN for feature extraction and a LSTM for temporal processing. Three CNN models are illustrated in the figure, including the baseline LeNet, and two variants of LeNet augmented with attention modules, one with STN and the other with DCN. The layers that handle geometric transformations are highlighted in the colored boxes. For the CNNs, conv is convolution layer, fc is fully-connected layer, pool is max-pooling layer. The numbers inside parentheses are: kernel size, number of output channels, and stride for conv and pool layers; number of output channels for offset2D layers; number of outputs for fc layers; size of output feature map for STN. The numbers inside the square brackets are trainable parameters in each layer given a 64×64 input image.

Attention modules

Spatial Transformer Networks (STN) [3] and Deformable Convolutional Networks (DCN) [1] are depicted in Fig. 2.

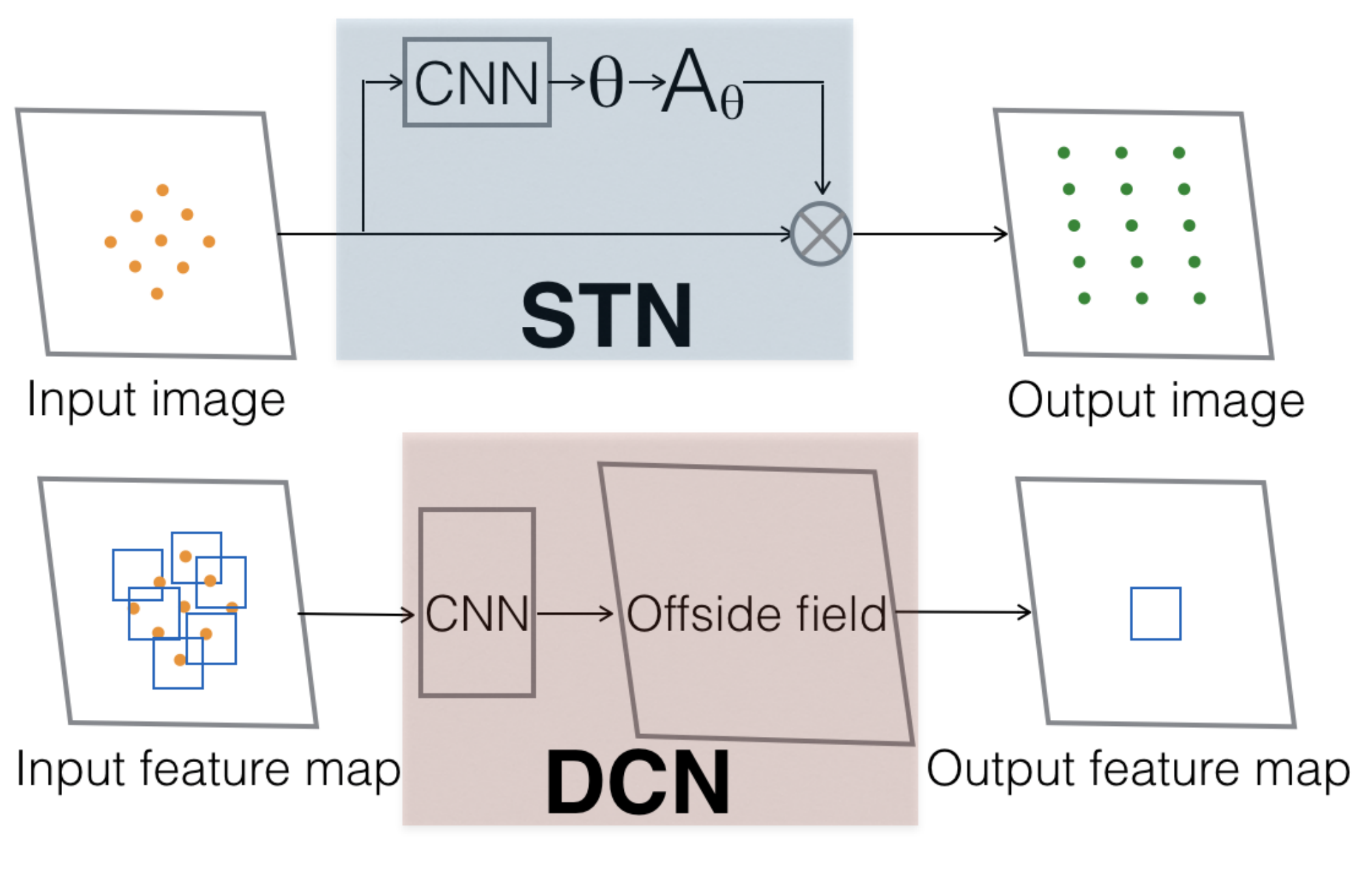

Figure 2: Structure of STN and DCN.

STN consists of three components: localization network, grid generator, and bilinear sampler. The localization network learns the affine transformation A_θ that should be applied to the image, i.e. $\theta = f_{loc}(\mathcal{I})$, where θ represents the transformation parameters. The grid generator produces coordinates $A_\theta(\mathcal{G})$ based on A_θ and a regular grid $\mathcal{G}$. The bilinear sampler takes input image and $A_\theta(\mathcal{G})$ to generate the output feature map.

In DCN, the spatial sampling locations in the convolution kernels are augmented with additional offsets learnt from classification. For example, a 5×5 kernel with dilation 1 in a convolutional layer samples on grid $\mathcal{R} = \{(-2, -2), (-2, -1), ..., (1, 2), (2, 2)\}$ on input feature map x.The feature at p_0 in output feature map y could be obtained based on Eq. (1a). The grid $\mathcal{R}$ is augmented by offsets $\{\Delta p_n | n = 1, ..., N\}, N = |R|$ in deformable convolution as in Eq. (1b), where the offset is implemented by bilinear interpolation.

$$y(p_0) = \sum_{p_n \in \mathcal{R}} w(p_n) \cdot x(p_0 + p_n) \tag{1a}$$

$$y(p_0) = \sum_{p_n \in \mathcal{R}} w(p_n) \cdot x(p_0 + p_n + \Delta p_n) \tag{1b}$$

Temporal processing

As shown in Eq. (2), Long Short Term Memory (LSTM) [2] contains a cell state c_t, which is controlled by input gate i_t, forget gate f_t, and output gate o_t. f_t decides which information to throw away, i_t determines what to store in c_t, while o_t generates the output based on the input x_t as well as the previous hidden state h_{t-1}. The W, b are the trainable parameters.

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{2a}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{2b}$$

$$g_t = \tanh(W_g \cdot [h_{t-1}, x_t] + b_g) \tag{2c}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \tag{2d}$$

Experiment

Dataset creation

This work proposes a synthetic dataset based on Moving MNIST [4], with augmentation that includes rotation and scaling. Sample images in the dataset are displayed in Fig. 3. Essentially, the video classification in this context is to recognize the digits in the presence of transformation and occlusion.

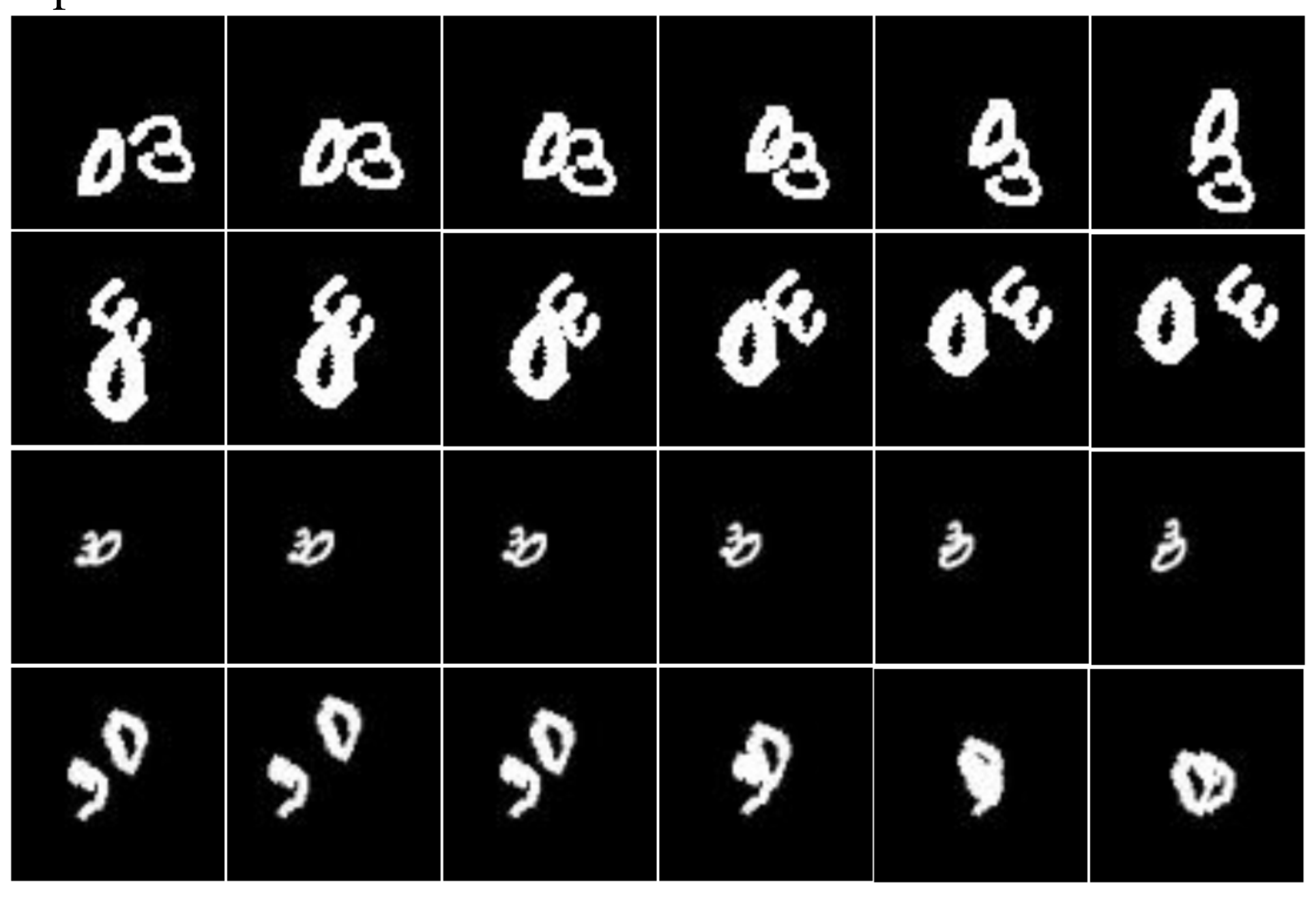

Figure 3: A sequence of images in class 03 in augmented Moving MNIST dataset. From first row to last row: original images, rotated images, scaled images, rotated and scaled images.

Training details

1. The models are pre-trained on the Scaled MNIST dataset
2. There are 100 classes in train and test set, and 10 different rotations or scales in each class, with 5 frames for one setting
3. Epochs: 10. Optimizer: RMSProp. Learning rate: 1×10^{-3}. Batch size: 50. Loss function: categorical cross entropy loss
4. LSTM is trained for 300 rounds to reduce the variance

Quantitative results on Moving MNIST

The cross entropy loss and test accuracy are shown in Table 1. The proposed model STN-LSTM and DCN-LSTM are compared against LeNet-LSTM as baseline.

Table 1: Comparison of cross entropy loss and test accuracy for the proposed model and baseline.

Moving MNIST	LeNet-LSTM	STN-LSTM	DCN-LSTM
Normal	$1.44, 97.96\%$	$1.98, 87.26\%$	$1.27, 99.62\%$
Rotation	$1.42, 98.43\%$	$1.97, 90.47\%$	$1.29, 99.70\%$
Scaling	$1.52, 96.28\%$	$1.99, 86.90\%$	$1.28, 99.41\%$
Rotation+Scaling	$1.51, 96.82\%$	$1.99, 89.10\%$	$1.25, 99.46\%$

1. DCN-LSTM consistently performs the best
2. STN-LSTM performs poorly since there are two digits in each image and STN performs affine transformation globally

Qualitative analysis on Moving MNIST

From the images processed by STN shown in Fig. 4, it could be observed that all the images are centered and transformed to a canonical orientation.

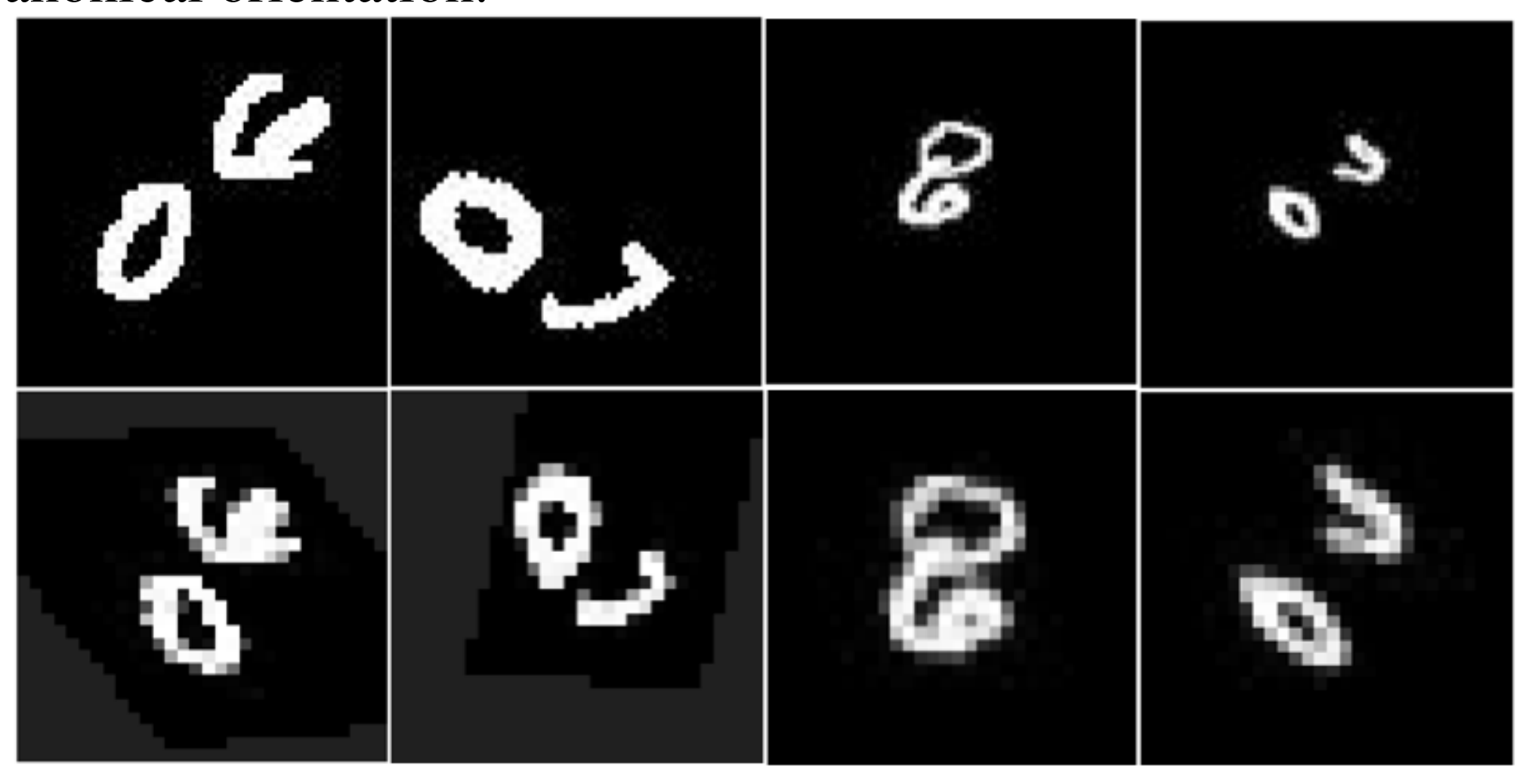

Figure 4: Output images of STN for class 06. First row: input images. Second row: output images of STN. From first column to last column: original images, rotated images, scaled images, rotated and scaled images.

Classification of digit gesture

Digit gesture has been widely used for evaluating the recognition algorithms, and classification of digit gesture is more challenging when the digits are written in the air by hand. Elastic deformation is applied to the images in Moving MNIST dataset, with rotation and scaling augmentation, to simulate the oscillations of the hand muscles. A sample image after applying elastic deformation is shown in Fig. 5.

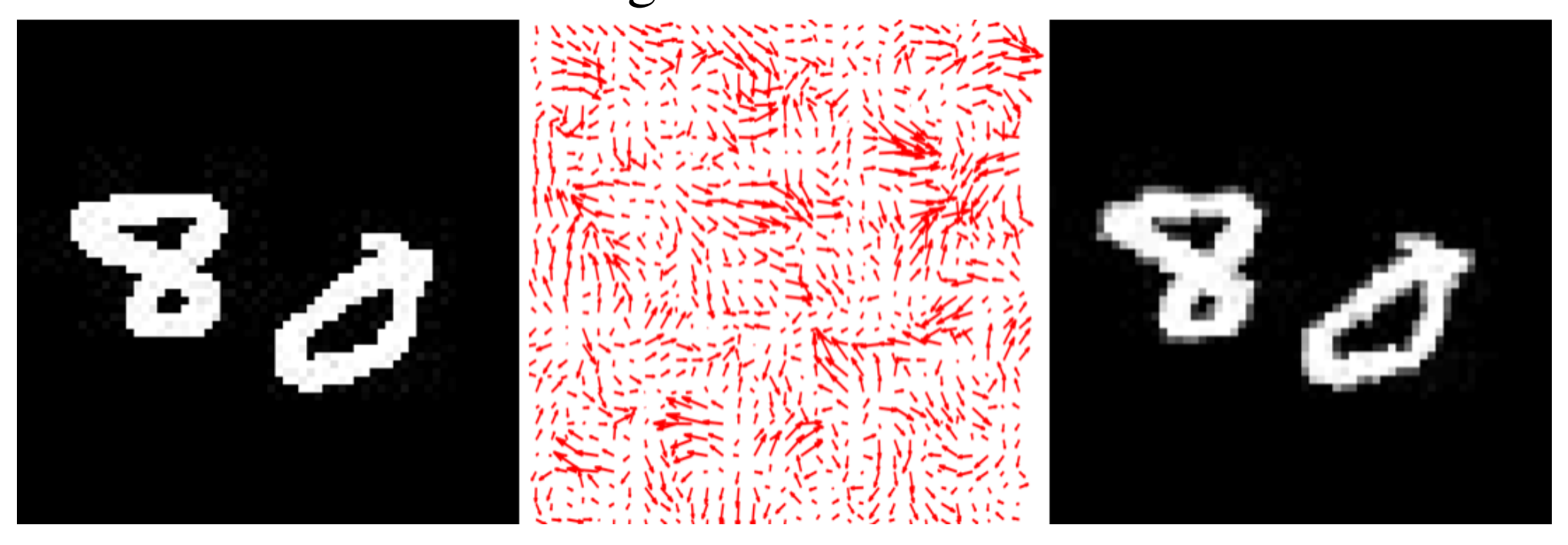

Figure 5: Image distorted by elastic deformation. From left to right: original image, elastic deformation field, deformed image.

The cross entropy loss and accuracy are $1.48, 97.19\%$ for LeNet, $1.48, 97.19\%$ for STN, and $1.28, 99.30\%$ for DCN, indicating that DCN-LSTM is still the best model to deal with distorted digits because of its deformable convolutional layers. Deformable Parts Models (DPM) could be formulated as CNNs; similarly, the deformable convolutional layers in the DCN-LSTM could be modified to explicitly learn the deformation field, which is better than DPM for certain classes. Therefore, the capability of the proposed model in recognizing digit gesture suggests that the model could be extended for tracking hand trajectory or dealing with multi-part articulated objects such as hand.

Key insights

1. DCN-LSTM achieves high accuracy compared to baseline
2. Attention is useful to deal with rotation and scale changes
3. STN-LSTM performs poorly due to global transformation
4. Future work: how to train the entire model end to end

References

[1] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. *arXiv preprint arXiv:1703.06211*, 2017.

[2] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[3] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *Advances in Neural Information Processing Systems*, pages 2017–2025, 2015.

[4] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhudinov. Unsupervised learning of video representations using lstms. In *International Conference on Machine Learning*, pages 843–852, 2015.