CS 574 Computer Vision using Machine Learning

Phase 2 Report

Spatial Transformer Networks
Group 16

***A**uthors:*

Desh Raj
130101018

Sumeet Ranka
130101062

Siddharth Kumar
130101072

Akashdeep Goswami
130101088

Samyak Kumbhalwar
130101041

Submitted: September 13, 2017

1 Introduction

In this report, we discuss the experiments conducted by our group in partial fulfillment of the second phase of the project on Spatial Transformer Networks. Specifically, we will briefly discuss the theory proposed by Jaderberg et al. in [1], and then describe the series of experiments performed to validate the theory.

After discussing the performed experiments and their results, we propose a new application for STNs in the form of image captioning problem. The rationale behind this proposal is explained in Section 4.

2 Background and Theory

In this section, we briefly describe the architecture of a spatial transformer network. The spatial transformer mechanism consists of three modules as shown in Fig 1.

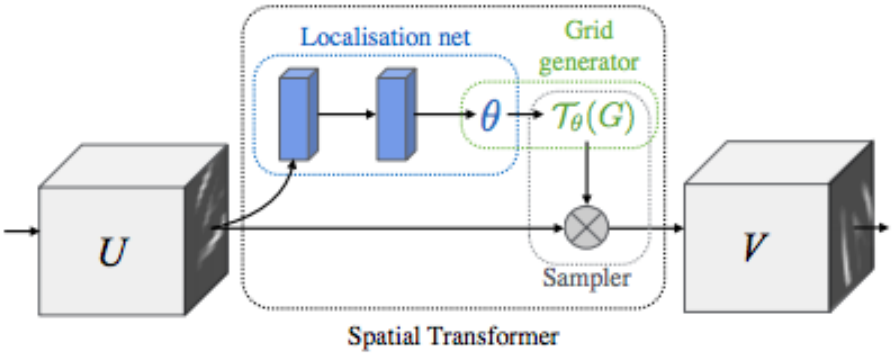


Fig. 1. The architecture of a spatial transformer module.

1. *Localisation network*: It takes the feature map as input and outputs the parameters of the spatial transformation that should be applied to it.
2. *Grid generator*: It uses the predicted transformation parameters to create a sampling grid, which is a set of points where the input feature map should be sampled to obtain the transformed image.
3. *Sampler*: It generates the image from the input feature map and the sampling grid.

The class of transformations τ_θ may contain different number of parameters, such as six in affine, eight in plane projective, piece-wise affine, or thin-plate spline. Further, the sampling kernel used in the sampler may be selected from a large number of available choices, such as an integer or a bilinear sampling kernel.

Jaderberg et al. use bilinear sampling for the STN in a CNN model consisting of two max-pooling layers. The model is trained using backpropagation with stochastic gradient descent, with three weight layers in the classification network.

3 Experiments

A series of experiments were performed, part of which involved implementing the paper. Other experiments were conducted for comparing results obtained from STN with those obtained from naive network models, or for improving the performance of the basic STN architecture by constructing variants with tuned parameters.

Each of the experiments was performed on Nvidia GPUs using the open-source Tensorflow framework. Source codes and data may be downloaded from Github[1].

3.1 Simple vs Distorted MNIST

In this series of experiments, we intend to emphasize the effect of distortion in the form of scaling, translation, rotation and cluttering, on the performance of a neural network model in image classification problems. For this purpose, we choose the popular MNIST handwritten character recognition dataset [2], and a distorted form of the same data generated by applying random transformations to the image.

Experiments were performed using two models of CNN: basic and complex, using both the simple and distorted data sets.

3.1.1 Basic CNN model

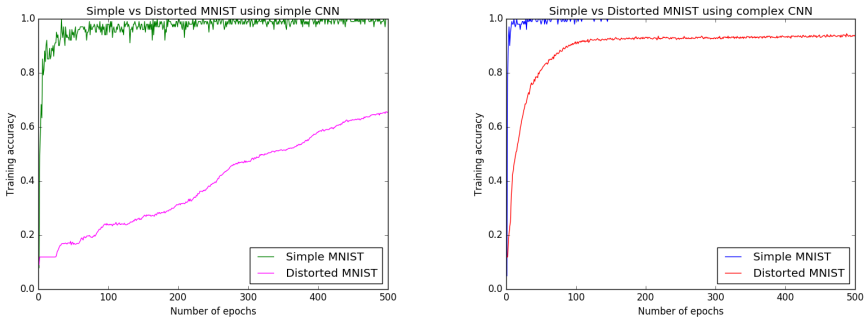
This model consists of 2 convolutional layers with 3x3 kernels (stride of 2), each followed by a 2x2 max pooling layer. Each of these convolutions generates 16 feature maps. Finally we have a 1024-node dense layer and a softmax layer to output class-wise probabilities.

The training accuracy of the model for 500 epochs on both the data sets is shown in Fig. 2(a). It is clear from the figure that distortion can reduce the performance of a network hugely.

3.1.2 Complex CNN model

This model consists of 2 convolutional layers with 5x5 kernels (stride of 1), each followed by a 2x2 max pooling layer. Each of these convolutions generates 32

[1] https://github.com/desh2608/CS574-CVML



(a) **(b)**

Fig. 2. Comparison of training accuracies with simple and distorted MNIST data for (a) basic CNN and (b) complex CNN.

feature maps. Finally we have a 1024-node dense layer and a softmax layer to output class-wise probabilities.

The training accuracy of the model for 500 epochs on both the data sets is shown in Fig. 2(b). As is clear from the figure, while the model performs significantly better than the basic CNN on the distorted MNIST, its accuracy is still less than that of the basic CNN on simple MNIST, even though its training time was about 5 times that of the other.

3.2 CNN vs CNN-STN on distorted MNIST

In this experiment, we implement the CNN-STN model used by the authors in [1]. The model is similar to the basic CNN model of the previous section, with the exception that the STN module is plugged in before the first convolutional layer.

To demonstrate the efficiency of the STN module, we perform the same experiment after removing the STN, on the same data and for the same number of iterations. This comparison is pictorially represented in Fig. 3.

From the figure, it can be observed that the STN module causes the training accuracy to cross the 90% threshold within 10-15 epochs and it stabilizes at around 95% before 100 epochs. On the other hand, a simple CNN model learns slowly, stabilizing at a low 80% accuracy after 200 epochs.

3.3 Improving CNN-STN on distorted MNIST

In Section 3.2, the training accuracy obtained on distorted MNIST data using the STN model was approximately 96%. Also, in Section 3.1, we saw that proper selection of kernel size, stride length and max pooling can greatly affect the performance of the network. This is also illustrated in Fig. 4.

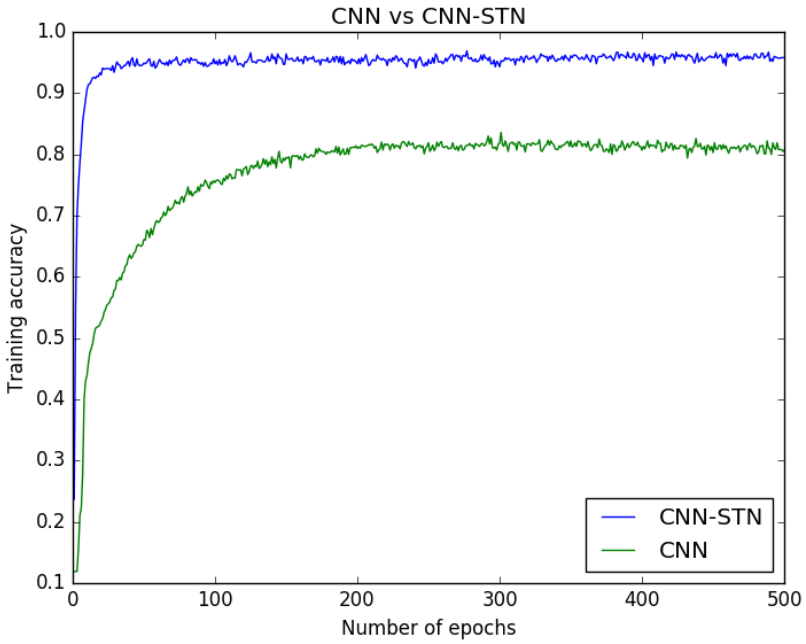


Fig. 3. Performance of the CNN-STN and simple CNN on distorted MNIST.

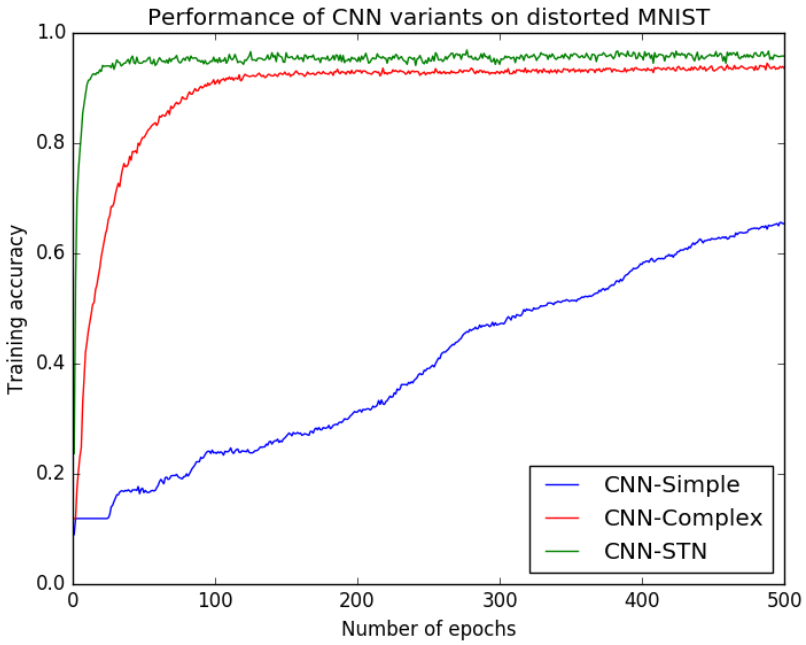


Fig. 4. Performance of CNN variants on distorted MNIST.

However, the performance of CNN-STN mentioned in the original paper is closer to 98-99% than the 96% we were obtaining using the vanilla model. So we performed another series of experiments by tweaking various features in the CNN-STN. These are described as follows.

3.3.1 STN Variant using complex CNN

In this model, we add an STN module to a complex CNN architecture as described earlier. The accuracy of the model on validation set with each epoch is shown by the "Variant 1" curve in Fig. 5.

3.3.2 STN Variant with variation localization network

In the original STN module, the localization network consists of 2 fully-connected layers. We perform experiments by modifying this in two ways:

1. First, we replace the first dense layer with a convolutional layer consisting of 5x5 kernel (stride of 1), and 32 feature maps as opposed to the 20 nodes in the fully-connected layer. Even though the model trained much faster than the original STN, it reached stagnation at around 94-95% accuracy, which is less than the original STN, as shown by the "Variant 2" curve in Fig. 5. We attributed this to the removal of the dense layer.

2. With the observations from the previous experiment, we hypothesized that adding another dense layer to the localization network may produce better classification results. However, the actual results obtained were disappointing, since this network also reached stagnation at around the same mark as Variant 2.

From point 2 above, we believe that having 2 dense layers in the localization network is optimum for the STN module. Further improvements, if any, may be obtained by either changing the model in which STN is used (as in Variant 1), or by tuning the parameters of the sampler. We propose to perform experiments to validate the latter idea in the next phase of the project.

3.4 Additional experiments on more complex data

In the Appendix section of their paper, the authors performed several other experiments using STN networks: MNIST addition, co-localisation, higher dimensional transformation, street view house number recognition, and fine-grained bird image classification. Among these, the MNIST addition and co-localisation involve multiple channels to the STN, and the higher dimensional transform requires multiple transformation parameters. This makes these problems very computationally-intensive and would require long training times. The model

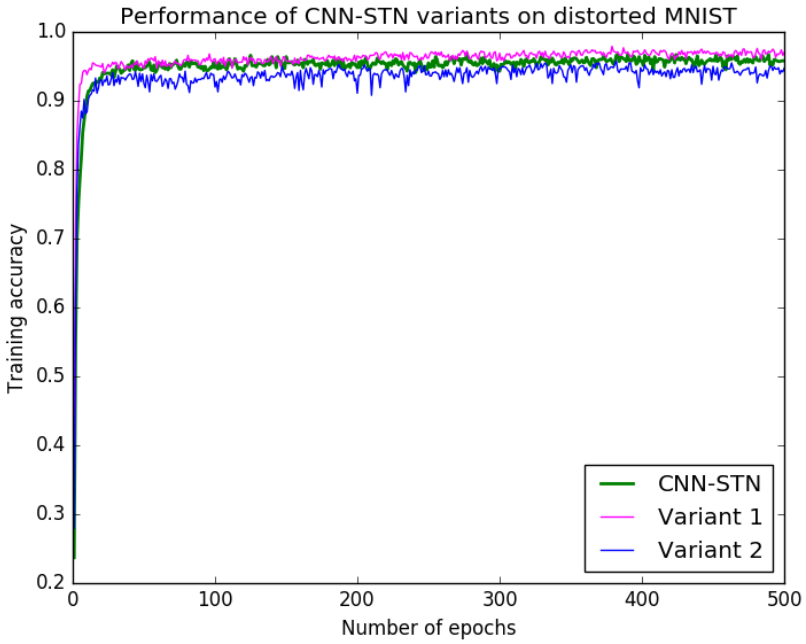


Fig. 5. Performance of STN variants on distorted MNIST.

used for fine-grained bird image classification uses the Inception network, which is quite an advanced architecture in itself. So in this section, we implement the model used for the experiment on Street View House Number recognition using STN.

3.4.1 Street View House Number

In this section, we try to classify the Street View House Number (SVHN) data, which has 2 formats: (i) the original 5-digit format, and (ii) the scaled, single-digit format. In their paper, the authors construct a very deep network with several layers in the STN, and numerous convolutional and parallel dense layers afterwards, to classify the first format. However, since we are short on computational resources, we use the second format, which consists of 100k 32x32 images with single digit.

First, we perform classification using a CNN model without the STN module. The accuracy on batches of the test set with every 10 iterations of training, is shown in Fig. 6. It can be observed that the accuracy is very low, which may be due to the low resolution of the images.

We are in the process of experimenting using a simpler version of the CNN-STN model than the one used in the papers. In essence, our model consists of 2 convolutional and max pooling layers followed by 2 dense layers in the lo-

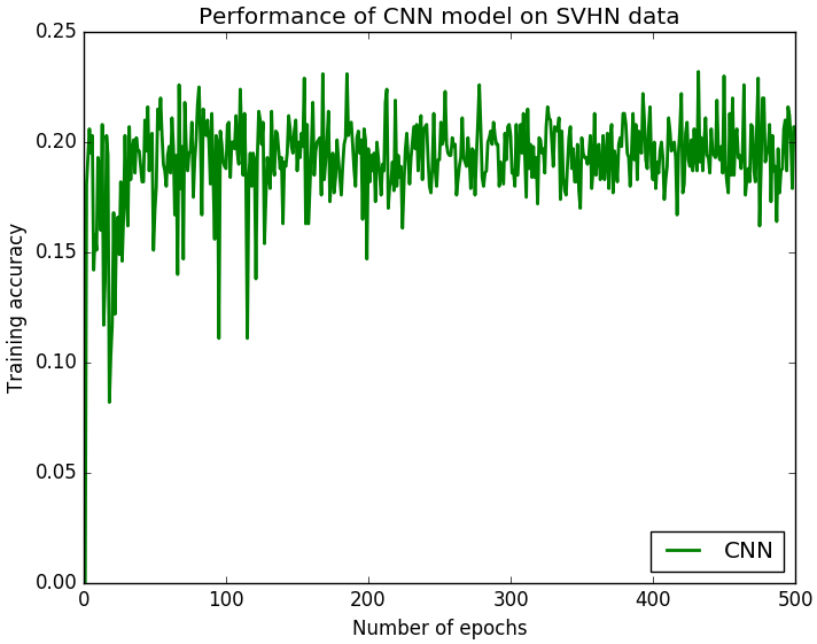


Fig. 6. Performance of CNN on SVHN data.

calisation network. The CNN consists of 3 convolutional layers followed by a fully-connected layer and a softmax layer. Each convolution uses ReLU activation, while the dense layers use non-linear activation, i.e. tanh or sigmoid.

4 Proposed improvement/application

In [1], the localisation network used consists of feedforward convolution layers ending with a max pooling layer. In [3], the feedforward network is replaced with a recurrent model for predicting the transformation matrices.

We earlier proposed a novel architecture that allows multiple glimpses over the input image and using the current glimpse as input to the RNN network. This is accomplished by extending the recurrence to the entire STN module instead of just the localisation network. Through this improvement, we expect to make the recognition task more accurate, especially with images consisting of multiple digits.

However, due to possible resource limitations in implementing such a model, we now propose to use the CNN-STN architecture for image captioning problems, since they have only been used for image classification up until now. We first give an overview of the image captioning problem and briefly describe the present approaches to solving it.

4.1 Overview

Automatically describing the content of an image is a fundamental problem in artificial intelligence that connects computer vision and natural language processing [4]. A description must capture not only the objects contained in an image, but it also must express how these objects relate to each other as well as their attributes and the activities they are involved in.

4.2 Current approaches

The popularity of CNN and neural language models led to a substantial increase in the number of models proposed for image captioning problem.

Kiros et al. [5] was the first to propose a neural network based approach for generating text descriptions from image and for image retrieval from text. They used a multimodal log bilinear model that was biased by the input image features.

A similar approach was taken by Mao et al. [6]. However, a recurrent model was used instead of the feed-forward neural language model. Vinyals et al. [4] proposed a model that made use of CNN for image feature generation. These features are then passed to an LSTM network.

All of the existing image captioning approaches can be classified into two types i.e top-down and bottom-up. The bottom-up approach starts with words describing the image and then combines them. The top-down approach starts from a gist of an image and then converts it into words. You et al. [7] made use of a novel semantic attention model which combines the visual information in both the approaches, top-down and bottom-up, in RNN framework.

In recent works, attention-based networks have been used for captioning with great success, and are expected to improve further with

4.3 Our proposal

In [1], the authors state thus: "A spatial transformer can be used for tasks requiring an attention mechanism, ..., but is more flexible and can be trained purely with backpropagation without reinforcement learning. A key benefit of using attention is that transformed (and so attended), lower resolution inputs can be used in favour of higher resolution raw inputs, resulting in increased computational efficiency."

For this reason, we believe that since image captioning uses attention as an input to the STN, replacing this attention layer with an STN module may be beneficial. Hence, we propose to implement STNs on image captioning problems.

References

[1] M. Jaderberg, K. Simonyan, A. Zisserman, et al., Spatial transformer networks, in: Advances in Neural Information Processing Systems, 2015, pp. 2017–2025.

[2] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86 (11) (1998) 2278–2324.

[3] S. K. Sønderby, C. K. Sønderby, L. Maaløe, O. Winther, Recurrent spatial transformer networks, arXiv preprint arXiv:1509.05329.

[4] O. Vinyals, A. Toshev, S. Bengio, D. Erhan, Show and tell: A neural image caption generator, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3156–3164.

[5] R. Kiros, R. Salakhutdinov, R. S. Zemel, Multimodal neural language models., in: ICML, Vol. 14, 2014, pp. 595–603.

[6] J. Mao, W. Xu, Y. Yang, J. Wang, Z. Huang, A. Yuille, Deep captioning with multimodal recurrent neural networks (m-rnn), arXiv preprint arXiv:1412.6632.

[7] Q. You, H. Jin, Z. Wang, C. Fang, J. Luo, Image captioning with semantic attention, arXiv preprint arXiv:1603.03925.