DLLAB WEEK 3 Pre-Report : Spatial Transformer Networks

Hyunsoo Cha[1], Kwanghoon Sohn[2]
[1]Yonsei University. [2]Yonsei University.

Convolutional Neural Networks do not have spatially diverse input data. Introducing the new learnable module, which is called the Spatial Transformer. This allows networks to manipulate data spatially within an explicit network. These different modules can be applied to the current convolutional architecture. It can be added in feature map itself without additional training supervision or modification of the optimization process. By using the spatial transformer, the researchers will show results for the model's translation, scale, rotation, and a model that learns the invariance of more warping.

The development direction so far has been the adaptation of the fast, scalable, and end-to-end learning framework. In general, due to the small spatial assistance for max-pooling, this spatial invariance was implemented only in the deep hierarchy and convolutions of max-pooling. These limitations of CNN are limited, and are due to pooling mechanisms to handle the theorem of already declared spatial data.

In this paper, researchers introduce the spatial transformer module. This is accompanied by proper behavioral learning during training for testing in questions without additional supervision. Unlike the pooling layer, the receptive field is fixed and peripheral, and the spatial transformer module is a dynamic mechanism, which provides a suitable conversion for each input sample, enabling spatial translation of feature maps and images. These transformations are carried out with a non-lateralized full feature map, which also includes scaling, cutting, rotating, and non-strict variations. This translates the network, including spatial transformation, not only to the selected region of the most relevant image, but also to the expected posture, part of the simple recognition of the following layers. Specifically, it is trained with a standard back-profile that permits an end-to-end model with a spatial transformation.

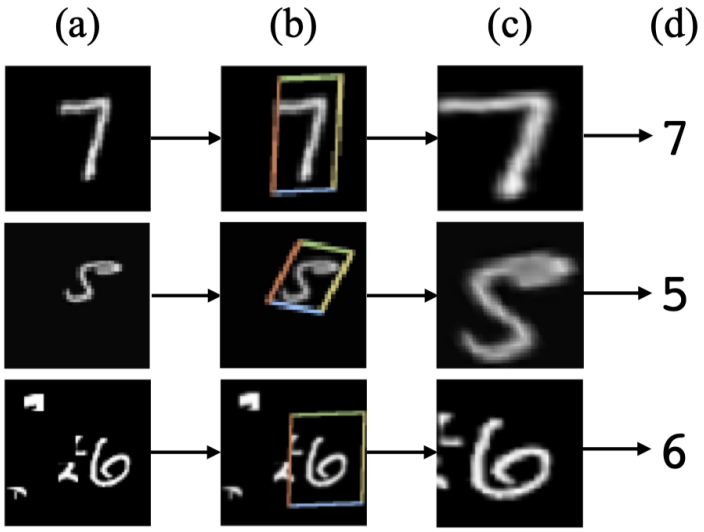

Figure 1: Example of a number of classification of distorted MNIST images using Spatial Transformer.[1]

Figure 1 discribes the result of classification of distorted MNIST datasets by using spatial transformers. It is (a) to display images of distorted MNIST numbers with random transformation, scale, rotation, and filling. (b) The peripheral (localization) network of spatial transformers expected the transformation to be applied to the input image. (c) The result of a spatial transformer. (d) is the expected number of classifications produced by the result of the spatial transformer followed by the full-connected network. CNNs, including spatial transformer modules, performed end-to-end training with only class labels. The groundtruth transformation did not have any knowledge given to the system. The network with spatial transformer just expected it as it is. In many ways, this test can benefit CNN. For example, there are three main types.

1. Image Classification: Suppose that CNN can train with multi-way in classifying images whether or not they contain a particular number. At this time, the space converter can cut and normalize the scale to simplify future classification. It also brings superior classification performance.

2. Co-Localisation: When configuring a set of images to contain the same class, but unknown class, the spatial transformer can be localized to each image.

3. Space Attention: spatial transformers are used for tasks that require an attention mechanism. However, it is more flexible and trained with back-propagation without reinforcement learning. The benefit of using attention is that it is transformed. Low-resolution inputs are used in support of high-resolution solutions.

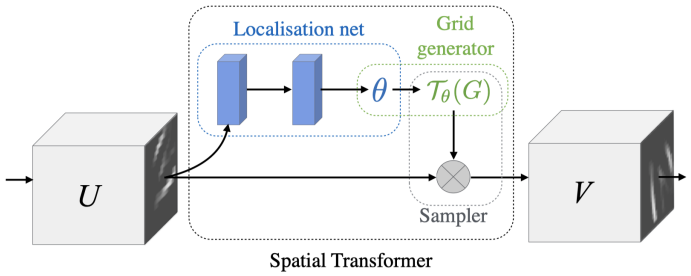

Figure 2: The structure of the space converter. Feature Map U passes through the local network. This returns to the conversion parameter θ. The standard spatial grid G,V is converted to the sampling grid $T_\theta(G)$. This sampling grid applies to U. This produces a output feature map V. A combination of local networks and sampling mechanisms constitute a spatial transformer.[1]

Let me explain the formula for the spatial transformer. Researchers consider a single transformation and a single output per transformer in this section. However, researchers also standardize multi-transformer in experiments. A detailed description of the above illustration is as in Figure 2. It consists of three main types. For computation, local networks accept input feature maps. Also, it should be applied through parameters of spatial transformation to feature maps through numbers in hidden layers. The predicted conversion parameters are then used to generate a sampling grid. This is a set of points, which must be created by an input map as a transforming output. This is also known as the grid generator. Finally, the feature map and sampling grid become the input of the sampler. The output map is sampled from the input to the grid point. A combination of these three elements creates a spatial transformer, which is described in detail in the following.

The local network takes the input feature Map $U \in \Re^{H\times W\times C}$. W means the width, H means the height, and C means the channel. In addition, the output is θ. There is also a T_θ conversion parameter that applies to feature map $\theta = f_{loc}(U)$. For example, Affine Transformation θ is six dimensions. The peripheral network function f_{loc} is possible in any form. However, it should include the last regression layer because the conversion parameter θ must be produced.

On a pixel-by-pixel basis, the researchers represented elements of a comprehensive feature map, not images needed. Comprehensively speaking, the output pixel is defined as being placed on the regular grid $G_i = (x_i^t, y_i^t)$ on the regular grid $G = G_i$. When a number of channels C, the width

W', and the height H' with the same input and output, there is a output feature map $V \in \Re^{H' \times W' \times C}$.

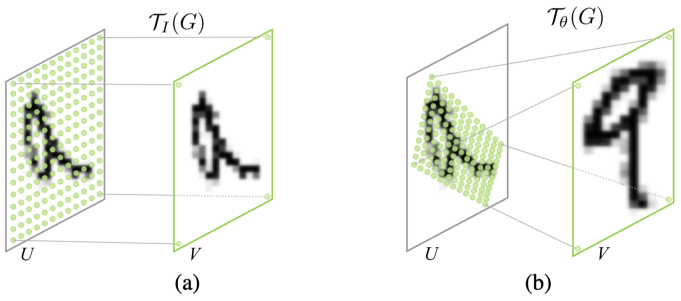

Figure 3: The above figure shows two examples of applying a parameterized sampling grid of the image U that produces the output V. (a) When I is the same conversion parameter, the regular grid $G = T_I(G)$ is called the sample grid. (b) Sample grid means the result of a regular grid warped with Affine Transformation $T_\theta(G)$.[1]

With the clarity of what is displayed, T_θ in an instant is assumed 2-dimensional affine transformation A_θ. The researchers will discuss other transformations. For affine cases, the pointwise transformation is as follows.

$$\begin{pmatrix} x_i^s \\ y_i^s \end{pmatrix} = T_\theta(G_i) = A_\theta \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix} = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \end{bmatrix} \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix} \tag{1}$$

(x_i^t, y_i^t) is the target coordinate of the regular grid on the output feature map. In addition, (x_i^s, y_i^s) is the source coordinate of the defined sample point, the input feature map. A_θ means an affine transformation matrix. The researchers use the height, width coordinates normalized in $-1 \leq x_i^t, y_i^t \leq 1$. This means within the spatial limits of the output. Similarly, $-1 \leq x_i^s, y_i^s \leq 1$ is defined within the spatial limit of the input, similar to y. Source and target conversion and sampling are the same as standard texture coordinates and mappings in graphics. The class of the transformation T_θ is forced. For example, let's note the affine transformation.

$$A_\theta = \begin{bmatrix} s & 0 & t_x \\ 0 & s & t_y \end{bmatrix} \tag{2}$$

This attention permits truncation, transformation, and isotropic scaling for the variables s, t_x, t_y. The transformation T_θ may also be more comprehensive. For example, it can be a plane projective transformation, thin plate spline, a piecewise affine, and 8 parameters. In practice, when the transformation is provided differently depending on the parameters, it can be in any parameterized form. This is the same point of the peripheral network output θ Significantly authorize backpropagated gradients from $T_\theta(G_i)$. In surrounding networks, complex tasks are often reduced, which is caused by low-dimensional methods and structural parameterization.

To perform spatial transformation of the input feature map, the sampler must import a set of sampling points $T_\theta(G)$. This is achieved by producing the output feature map V sampled along the input feature map U. Each $T_\theta(G)$ coordinate (x_i^s, y_i^s) determines that the sampling kernel takes the value of a particular pixel from the result V. As summary, researchers expressed the notion of sampling kernel to formula as follows:

$$V_i^c = \sum_n^H \sum_m^W U_{nm}^c k(x_i^s - m; \Phi_x) k(y_i^s - n; \Phi_y) \tag{3}$$

$$\forall i \in [1 \ldots H'W'] \forall c \in [1 \ldots C] \tag{4}$$

Φ_x, Φ_y is a parameter of the generic sampling kernel $k()$ that determines the interpolation image, such as bilinear. U_{mn}^c means the value from channel c to (m,n) coordinates. V_i^c is the result of the pixel i of the coordinate (x_i^t, y_i^t) on the channel c. By granting loss of backpropagation through sampling mechanisms, researchers can determine gradients for U and G. Partial differentiation by organizing the above formulas results in the following:

$$\frac{\partial V_i^c}{\partial U_{nm}^c} = \sum_n^H \sum_m^W max(0, 1 - |x_i^s - m|) max(0, 1 - |y_i^s - n|) \tag{5}$$

$$\frac{\partial V_i^c}{\partial x_i^s} = \sum_n^H \sum_m^W U_{nm}^c max(0, 1 - |y_i^s - n|) \begin{cases} 0 & if \quad |m - x_i^s| \geq 1 \\ 1 & if \quad m \geq x_i^s \\ -1 & if \quad m < x_i^s \end{cases} \tag{6}$$

This provides a sampling mechanism to discriminate against, granting input feature maps as well as loss gradients that follow the sampling grid coordinates.

Any point that enables a drop from a self-contained module to a CNN structure causes a spatial transformer network. These trigger points are created by a combination of spatial transformers, local networks, and grid generators. The module is computationally very fast and cannot slow down training. Because of subsequent downsampling, which can be applied to the output of the transformer, it can speed up to the active model and very little time is overheaded when originally used.

In conclusion, researchers can have multiple spatial transformers on CNN. Implementing multiple spatial transformers that increase the depth of the network increases the abstract representation and also allows the information representation based on the expected transformation parameters to be more potentially given to the local network. To perform modeling by limiting the number of objects, the number of parallel spatial transformers approaches the limit of this structure, which occurs in pure feedforward networks.

Experiments have been shown by using a spatially transformer network through a number of supervised learning tasks. The researchers used MNIST dataset as a testbed to explore the range of transformations that the network could learn from by using spatial transformers. The distortion was attempted in various ways, which was briefly shown in the following illustration.

Model		MNIST Distortion R	RTS	P	E
FCN		2.1	5.2	3.1	3.2
CNN		1.2	0.8	1.5	1.4
ST-FCN	Aff	1.2	0.8	1.5	2.7
	Proj	1.3	0.9	1.4	2.6
	TPS	1.1	0.8	1.4	2.4
ST-CNN	Aff	0.7	0.5	0.8	1.2
	Proj	0.8	0.6	0.8	1.3
	TPS	0.7	0.5	0.8	1.1

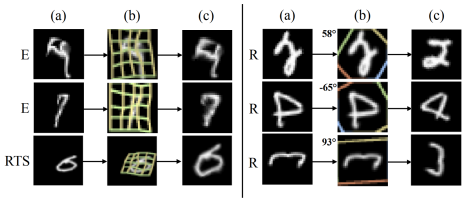

Figure 4: The left side of the figure represents the percentage error in the MNIST data distorted by each model. On the right hand side, CNN failed, but obtained several test images that were successfully classified by the spatial transformer network.[1]

Rotations are marked (R) and rotations, scale, and translation occur simultaneously (RTS). Projective transformation is expressed as (P). Finally, elastic warping is written as (E). The researchers trained baseline full-connected (FCN) and convolutional (CNN) neural networks. Spatial transformer networks used bilinear sampling, but variables used different transformation variables. Examples include affine transformation (aff), projective transformation (Proj), and thin plate spline transformation (TPS). All networks used include the same number of parameters. In CNN models, the maximum number of pooling layers is two. Inside all networks, optimization techniques also remain the same. The multinomial cross-entropy loss, SGD, backpropagation, and scheduled learning rate reduction use three weights. As Figure 4 shows, when observing a particular distorted type of data, the spatial transformer enables the network and becomes superior to the opposite base network. For RTS, ST-CNN achieves 0.5%. This is the result of relying on a class of transformations using T_θ. CNN, on the other hand, achieves 0.8 percent error in two max-pooling layers. Usually, a spatial transformer is a different way of achieving a spatial variation. In conclusion, the researchers introduced a module named Spatial Transformer. This module enters the network and externally performs spatial transformations of features.

[1] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transformer networks. *arXiv preprint arXiv:1506.02025*, 2015.