DEEP LEARNING LAB(EEE4423-01) Week 4 : STN(Spatial Transformer Networks)

Hyeon Si Eun[1], 2019142214
[1] Department of Electric and Electronic Engineering, Yonsei University

1 STN(Spatial Transformer Networks)

The paper "Spatial Transformer Networks" (STN)[1] proposes a new type of neural network layer called Spatial Transformer that can be inserted into CNN. It can learn to spatially transform input data in order to improve the performance of the network on a given task.

The authors of this paper argue that the intermediate feature maps (convolutional layer activations) in a CNN are not actually invariant to large transformations of the input data. Although CNN's max-pooling layer somewhat help to satisfy this property by allowing a network to be spatially invariant to the position of features, it is difficult to cope with the various spatial variability of the data with 2 X 2 pixel-wise operations. The spatial variability means spatial changes such as scale, rotation, and translation.

This new neural network architecture can dynamically manipulate the spatial relationship between inputs and outputs in a differentiable manner. This allows the network to learn to perform tasks that require spatial transformation, such as image recognition or object detection.

The authors of the paper noticed that many tasks in computer vision require spatial manipulation of the input data, such as rotating, scaling, or cropping an image. However, most neural networks do not explicitly account for spatial transformation and are thus limited in their ability to perform such tasks.

To address this limitation, the authors proposed a novel architecture called Spacial Transformer for neural networks that can learn to perform spatial transformations in a differentiable manner.

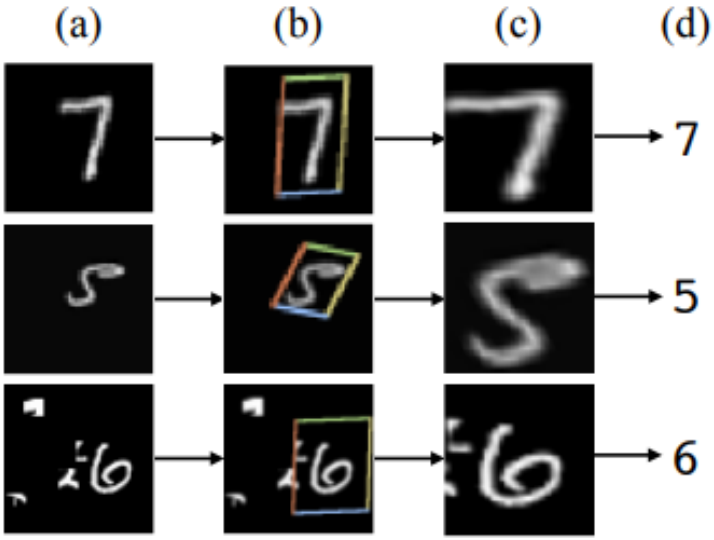

Figure 1: The figure is the result of using a spatial transformer right in front of a fully-connected network and training for MNIST digit classification. For the inputs in (a), each of which has a size, angle, and center position, the spatial transformer finds the rectangular area shown in (b) and creates a transformed output as shown in figure (c) and passes it to the fully-connected network. The result is the numeric value predicted by the classifier (d). The behavior of the spatial transformer is different for each input data sample and is learned during the learning process without special supervision. In other words, it is important that it is learned all at once through backpropagation during the end-to-end training process of the used model.

Spatial transformer takes an image or feature map as input and supports not only scaling, cropping, rotation, but also non-rigid deformation such as thin plate splines. It can be used to select only the most relevant areas of the current image, or to convert the current image or feature map to the most common canonical form to aid in the inference operation of the neural network layer that follows. This can be applied equally to multi-channel input. This is because even if it is called a multi-channel, it can be applied individually to each channel.

This is an extended abstract. The full paper is available at the this paper.

2 The structure of Spacial Transformer

The STN consists of three main components: a localization network, a grid generator, and a sampler.

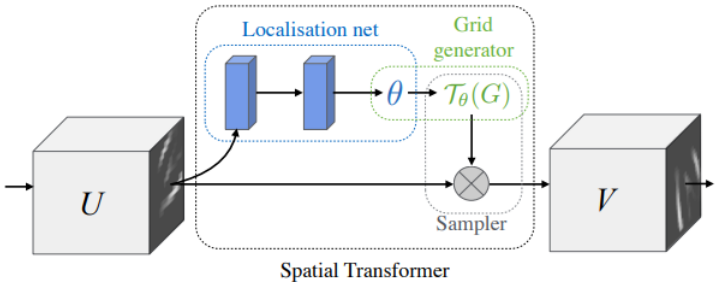

Figure 2: The architecture of a spatial transformer module.

The Localization Network estimates the parameter matrix θ of the transformer to be applied to the input feature map U.The localization network is a small neural network that takes an input image and outputs the parameters of an affine transformation that can be used to spatially transform the input data. This transformation includes rotation, translation, scaling, and shearing. The parameters of this transformation are learned by the STN during training, allowing it to learn the optimal transformation for a given task.

Grid Generator calculates sampling grid $\mathcal{T}_\theta(G)$, which determines the location of the point to be sampled on the input feature map according to the estimated θ. The grid generator takes the affine transformation parameters output by the localization network and generates a grid of sampling points. These points correspond to the location of pixels in the input image that will be used to produce the output image.

Finally, the sampler applies the sampling grid to the input feature map U to create the converted output feature map V. The sampler uses the grid generated by the grid generator to sample the input image and produce the transformed output image. The sampling process is differentiable, which means that the STN can be trained end-to-end using backpropagation.

3 A localization network

The Localization Network estimates the parameter matrix θ of the transformer to be applied to the input feature map $U \in R^{H\times W\times C}$ has a width W, length H, and number of channels C. Localization network can be applied to either fully-connected network or convolutional network, as long as there is a regression layer at the end to estimate the transformation parameter θ. In the experiments in this paper, the authors used four layer CNNs and two layer FNNs.

4 A grid generator

The Grid generator calculates the sampling grid $\mathcal{T}_\theta(G)$ that determines the location of the point to be sampled on the input feature map according to the estimated θ. Each pixel (x_i^t, y_i^t) of output V is located on top of a regular grid G, which is a typical rectangular shape. The sampling grid G of the

output V is mapped to the sampling grid $\mathcal{T}_\theta(G)$ of the input U via transform $\mathcal{T}_\theta$.

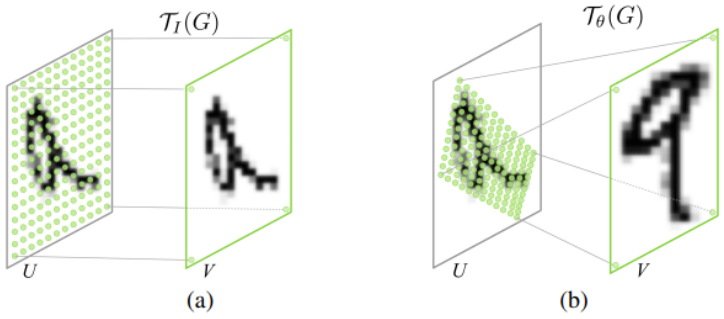

Figure 3: Two examples of applying the parameterised sampling grid to an image U producing the output V . (a) The sampling grid is the regular grid $G = \mathcal{T}_I(G)$, where I is the identity transformation parameters. (b) The sampling grid is the result of warping the regular grid with an affine transformation $\mathcal{T}_\theta(G)$.

For example, If $\mathcal{T}_\theta$ is 2D affine Transformation A_θ, we can express it as follows:

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

Affine transform can express scale, rotation, translation, skew, cropping with six parameters. As another example, the attention model, which represents isotropic scale, translation, and cropping in three parameters, can be represented by the formula below.

$$\begin{pmatrix} x_i^s \\ y_i^s \end{pmatrix} = \mathcal{T}_\theta(G_i) = \mathrm{A}_\theta \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix} = \begin{bmatrix} s & 0 & t_x \\ 0 & s & t_y \end{bmatrix} \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix} \tag{2}$$

As long as Transformation $\mathcal{T}_\theta$ is differentiable for each parameter, it can express all other common transformations such as projective transformation, thin plate split transformation, and so on.

5 A sampler

Sampler applies sampling grid $\mathcal{T}_\theta$ to input feature map U to create a converted output feature map V. In order to obtain a specific pixel value in output V, sampling grid $\mathcal{T}_\theta$ has the value from which position in the input U is obtained. Each $\left(x_i^t, y_i^t\right)$ coordinate in $\mathcal{T}_\theta(G)$ defines the spatial location in the input where a sampling kernel is applied to get the value at a particular pixel in the output V.

$$V_i^c, = \sum_n^H \sum_m^W U_{nm}^c k(x_i^s - m; \Phi_x) k(y_i^s - n; \Phi_y), \forall i \in [1 \ldots H'W'], \forall c \in [1 \ldots C] \tag{3}$$

The function that implements image interpolation is denoted by the general sampling kernel $k()$. Φ_x and Φ_y are the parameters of a generic sampling kernel. In the case of nearest integer interpolation, the equation is as follows:

$$V_i^c = \sum_n^H \sum_m^W U_{nm}^c \delta(\lfloor x_i^s + 0.5 \rfloor - m) \delta(\lfloor y_i^s + 0.5 \rfloor - n) \tag{4}$$

$\lfloor x_i^s + 0.5 \rfloor - m$ rounds x to the nearest integer and $\delta()$ is the Kronecker delta function. Alternatively, a bilinear sampling kernel can be used, giving

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

To calculate the loss value as backpropagation across the entire network, it must be differentiable for U and G. In the case of Bilinar interpolation, each partial derivative is calculated as follows:

$$\frac{\partial V_i^c}{\partial U_{nm}^c} = \sum_n^H \sum_m^W \max(0, 1 - |x_i^s - m|) \max(0, 1 - |y_i^s - n|) \tag{6}$$

$$\frac{\partial V_i^c}{\partial x_i^c} = \sum_n^H \sum_m^W U_{nm}^c \max(0, 1 - |y_i^s - n|) \begin{cases} 0 & if \quad |m - x_i^s| \geq 1 \\ 1 & if \quad m \geq x_i^s \\ -1 & if \quad m < x_i^s \end{cases} \tag{7}$$

similarly to (7) for $\frac{\partial V_i^c}{\partial y_i^c}$. Even if the sampling function is not differentiable on all intervals, backpropagation can be calculated by dividing it into intervals and using subgradients.

6 Spatial Transformer Networks

A spatial transformer module composed of a localization network, grid generator, and sampler inserted into a CNN structure is called a Spatial Transformer Network. Spatial transformer modules can theoretically be inserted at any point in the CNN, and any number of them.

How the spatial transformer transforms the input feature maps is learned during training to minimize the overall cost function of the CNN. Therefore, the authors claim that it has little impact on the overall training rate.

It is most common to place a Spatial Transformer module directly in front of the CNN's input, but it can also be placed in a deeper layer inside the network to apply it for more abstract information, or to track multiple parts of an image in parallel.

7 Experiments

The first experiment is an experiment to improve classification performance with a spatial transformer for the distorted MNIST dataset.

The second experiment is a digit recognition experiment on the Street View House Numbers (SVHN) dataset, which consists of 200,000 photos of real house address signs.

The third experiment is in which fine-grained bird classification is applied to Caltech's CUB-200-2011 birds dataset, which consists of 11,788 photos of 200 species of birds.

If you want to see detailed experimental results, we recommend viewing the full paper.

8 Finish

The authors demonstrate the effectiveness of the STN layer on a variety of tasks, including image classification, object detection, and geometric matching. They show that the STN can learn the optimal spatial transformation that would be difficult to achieve using handcrafted feature engineering. They also show that the STN layer can learn to perform a wide range of spatial transformations, including translations, rotations, scaling, and non-rigid deformations. In addition, The STN can be integrated into existing neural network architectures as a module, allowing it to be used in combination with other network components. The STN layer can be inserted into any neural network architecture and trained end-to-end along with the rest of the network.

Overall, the STN represents a significant advancement in the field of computer vision and deep learning, particularly in the area of spatial reasoning and manipulation. the STN layer provides a powerful and flexible mechanism for incorporating spatial transformations into neural network architectures, which can help to improve the performance of these models on a wide range of tasks.

[1] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. *Advances in neural information processing systems*, 28, 2015.