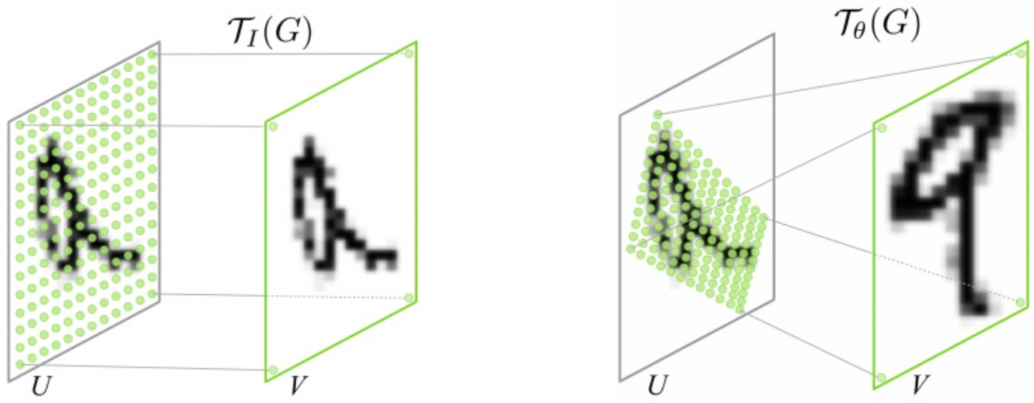
SPATIAL TRANSFORMER NETWORKS

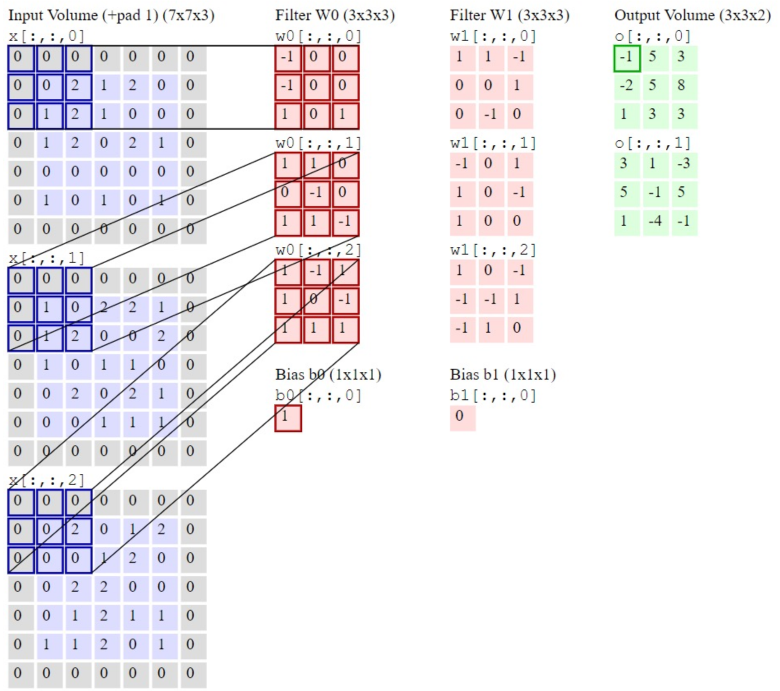
MAX JADERBERG, KAREN SIMONYAN, ANDREW ZISSERMAN, KORAY KAVUKCUOGLU
GOOGLE DEEPMIND, LONDON, UK



Motivation

- CNNs are prone to transformations (e.g. scaling and rotation)
  - https://www.cs.ryerson.ca/~aharley/vis/conv/flat.html

- Spatial transformer networks learn how to perform spatial transformations on the input image in order to enhance the geometric invariance of the model.



Input Volume (+pad 1) (7x7x3)

x[:,:,0]

0	0	0	0	0	0	0
0	0	2	1	2	0	0
0	1	2	1	0	0	0
0	1	2	0	2	1	0
0	0	0	0	0	0	0
0	1	0	1	0	1	0
0	0	0	0	0	0	0

x[:,:,1]

0	0	0	0	0	0	0
0	1	0	2	2	1	0
0	1	2	0	0	2	0
0	1	0	1	1	0	0
0	0	2	0	2	1	0
0	0	0	1	1	1	0
0	0	0	0	0	0	0

x[:,:,2]

0	0	0	0	0	0	0
0	0	2	0	1	2	0
0	0	0	1	2	0	0
0	0	2	2	0	0	0
0	0	1	2	1	1	0
0	1	1	2	0	1	0
0	0	0	0	0	0	0

Filter W0 (3x3x3)

w0[:,:,0]

-1	0	0
-1	0	0
1	0	1

w0[:,:,1]

1	1	0
0	-1	0
1	1	-1

w0[:,:,2]

1	-1	1
1	0	-1
1	1	1

Bias b0 (1x1x1)

b0[:,:,0]

1

Filter W1 (3x3x3)

w1[:,:,0]

1	1	-1
0	0	1
0	-1	0

w1[:,:,1]

-1	0	1
1	0	-1
1	0	0

w1[:,:,2]

1	0	-1
-1	-1	1
-1	1	0

Bias b1 (1x1x1)

b1[:,:,0]

0

Output Volume (3x3x2)

o[:,:,0]

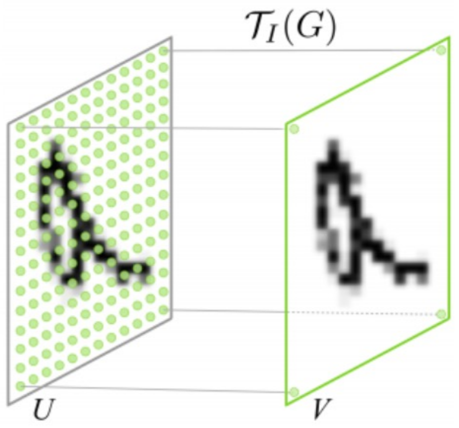
-1	5	3
-2	5	8
1	3	3

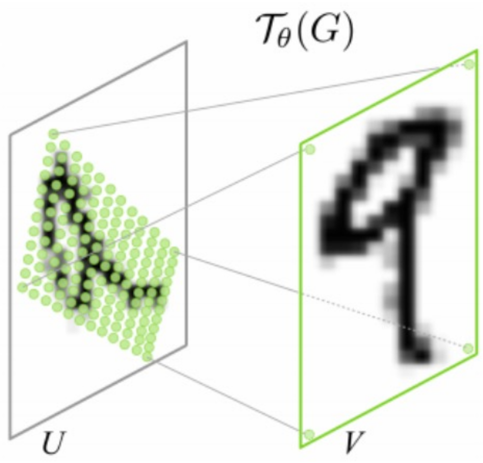
o[:,:,1]

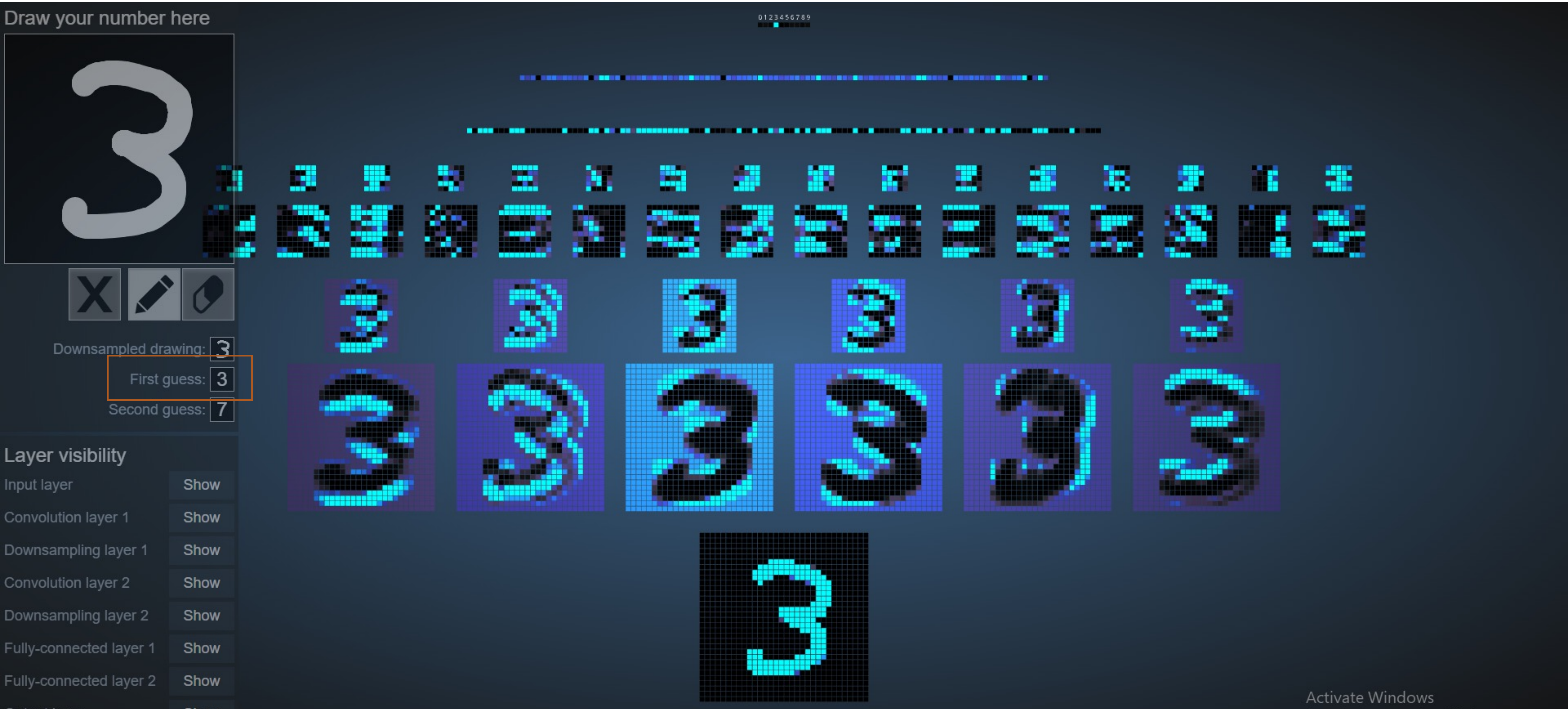
3	1	-3
5	-1	5
1	-4	-1

Spatial transformer networks

- Two examples of applying the parameterised sampling grid to an image U producing the output V:
  - Top: the sampling grid is the regular grid G = TI(G), where I is the identity transformation parameters.
  - Bottom: the sampling grid is the result of warping the regular grid with an affine transformation Tθ(G).







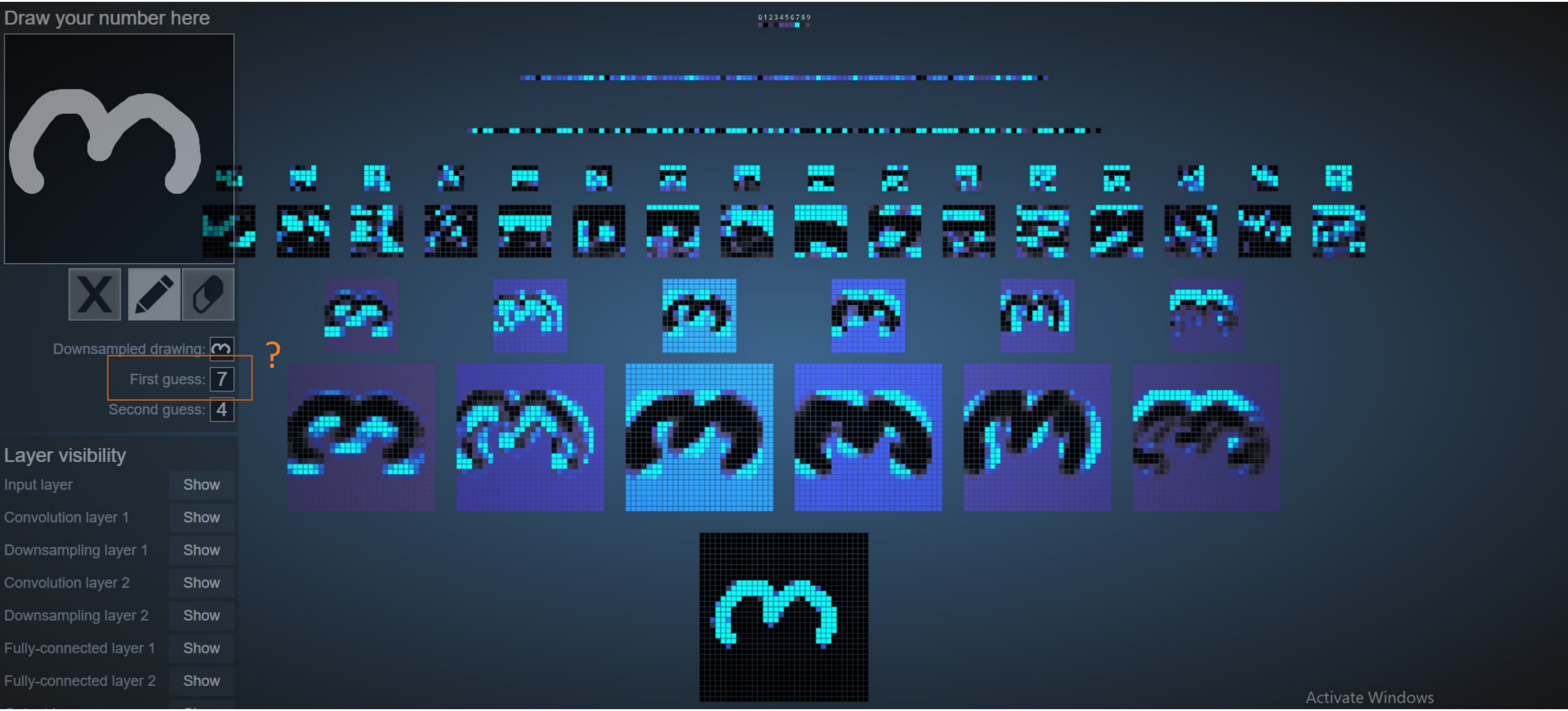
Downsampled drawing: 3

First guess: 3

Second guess: 7

Layer visibility

Input layer	Show
Convolution layer 1	Show
Downsampling layer 1	Show
Convolution layer 2	Show
Downsampling layer 2	Show
Fully-connected layer 1	Show
Fully-connected layer 2	Show



Draw your number here

Downsampled drawing:

First guess: 7

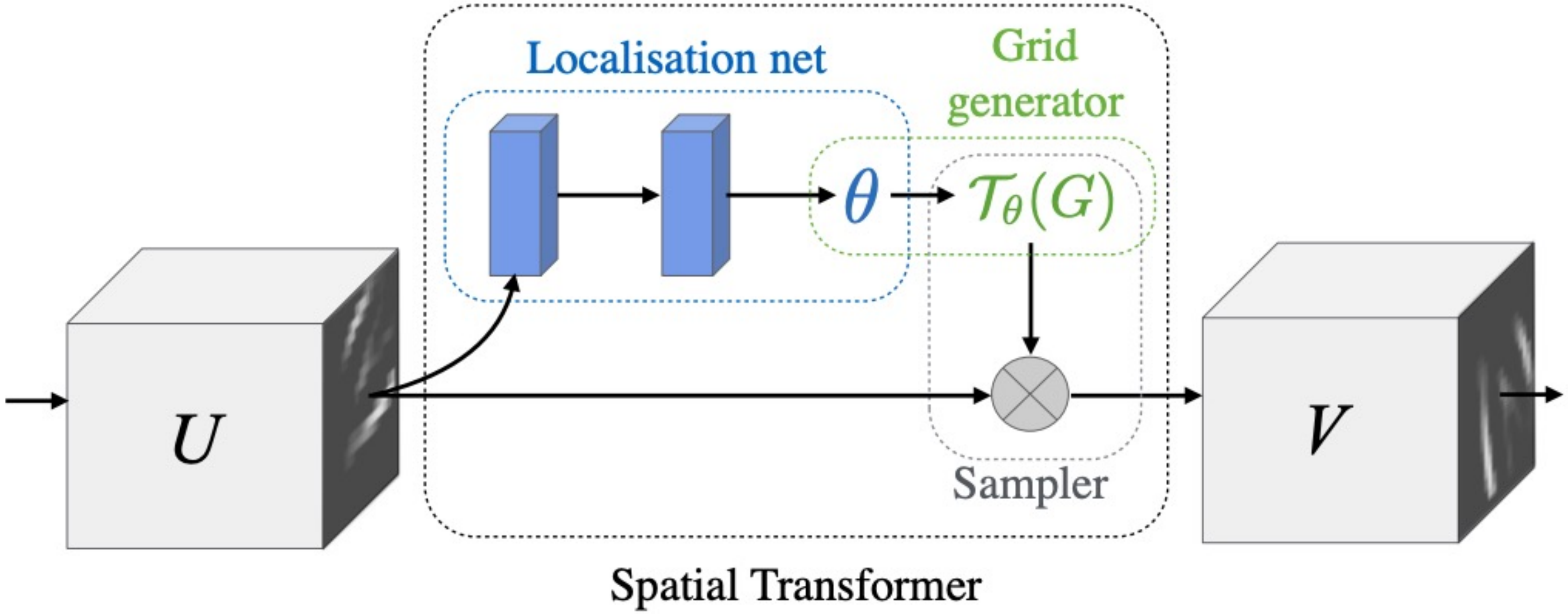
?

Second guess: 4

Layer visibility

Input layer	Show
Convolution layer 1	Show
Downsampling layer 1	Show
Convolution layer 2	Show
Downsampling layer 2	Show
Fully-connected layer 1	Show
Fully-connected layer 2	Show

Spatial transformer networks: components

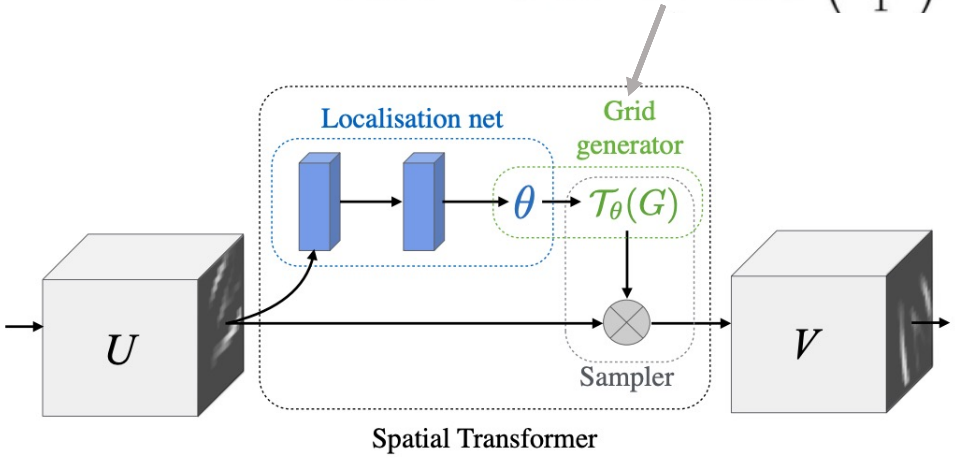


Spatial transformer networks: components

- **Grid generator:**
  - A transformation in which we aim to apply is first defined (e.g. rotation or translation). This can be represented as a mathematical expression (matrix multiplication).

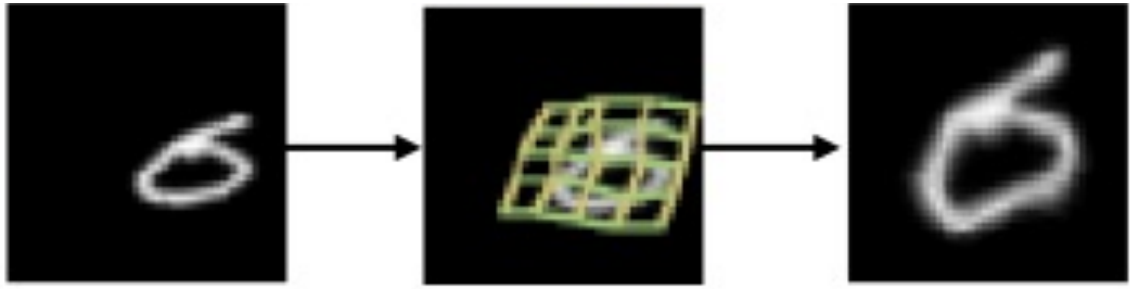
Grid generator uses θ to compute sampling grid

$$\begin{pmatrix} x_i^s \\ y_i^s \end{pmatrix} = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \end{bmatrix} \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix}$$



Spatial transformer networks: components

- **Grid generator:**
  - Grid generator generates a grid of coordinates in the input image corresponding to each pixel from the output image.

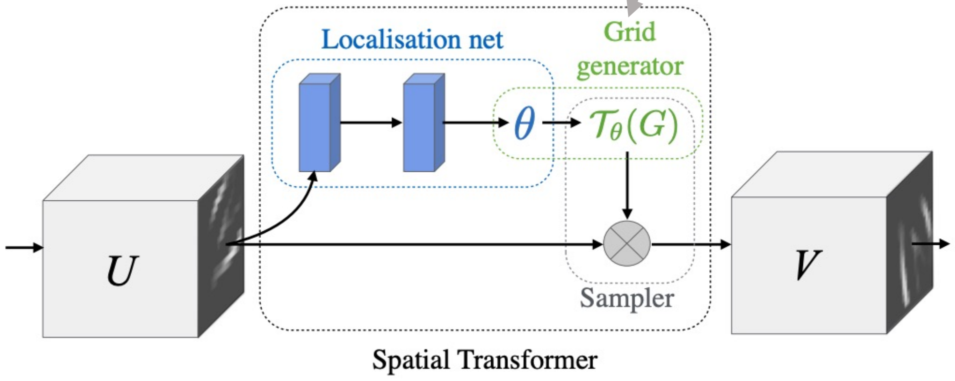


Spatial transformer networks: components

- **Grid generator:**
  - How do we choose the transformation parameters to use/apply in the grid generator (i.e. Tθ(G))?

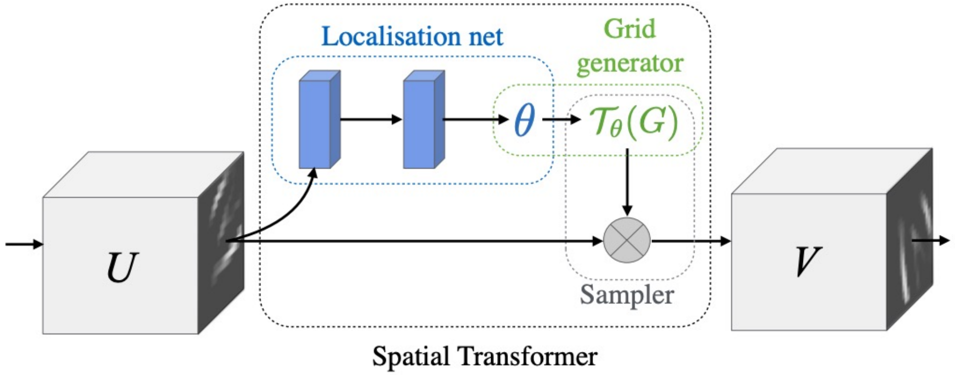
Grid generator uses θ to compute sampling grid

$$\begin{pmatrix} x_i^s \\ y_i^s \end{pmatrix} = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \end{bmatrix} \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix}$$



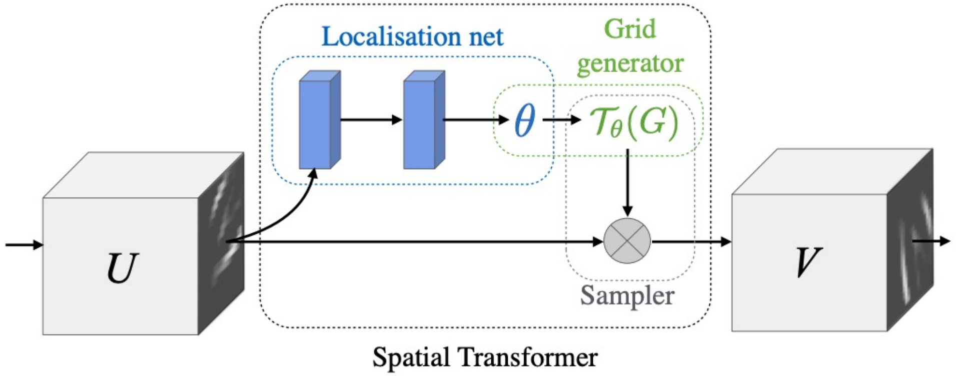
Spatial transformer networks: components

- **Localization network:**
    - Predicts the transformation theta.
        - i.e. regresses the transformation parameters.
    - Note: This is not explicitly learned from this dataset, in fact, the network learns the spatial transformations that optimizes the overall performance on the given task.

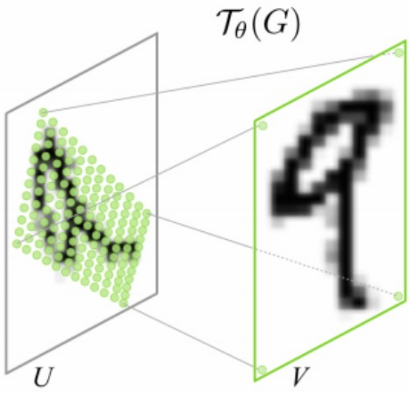


Spatial transformer networks: components

- **Sampler:**
  - Uses the parameters of the transformation and applies it to the input image.
  - Sampled values are computed using some interpolation.



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

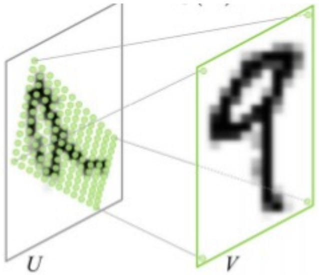


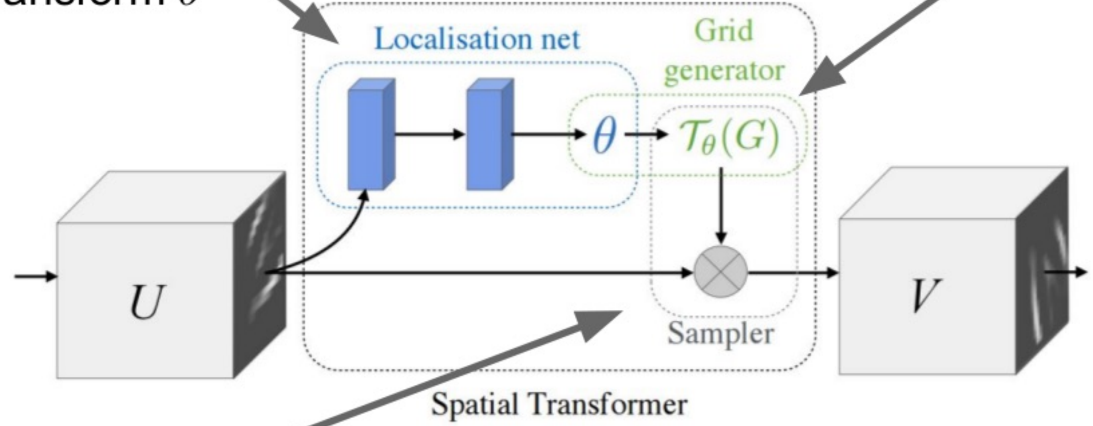
Spatial transformer networks: components

A small **Localization network** predicts transform θ

Grid generator uses θ to compute sampling grid

$$\begin{pmatrix} x_i^s \\ y_i^s \end{pmatrix} = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \end{bmatrix} \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix}$$





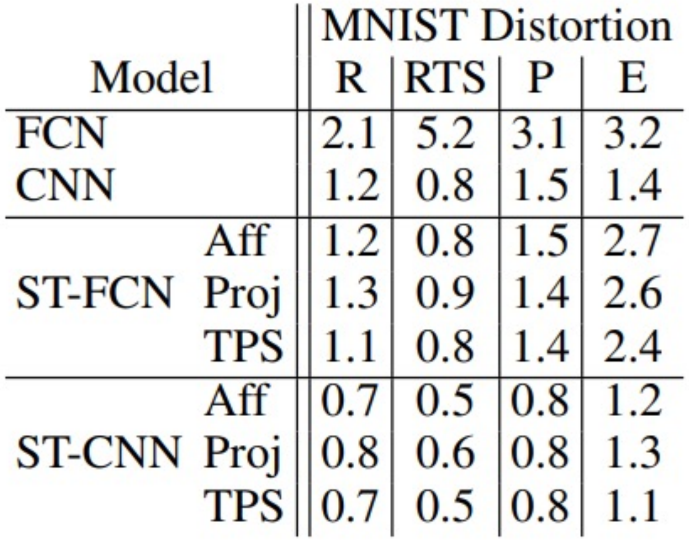
Input: Full image

Output: Region of interest from input

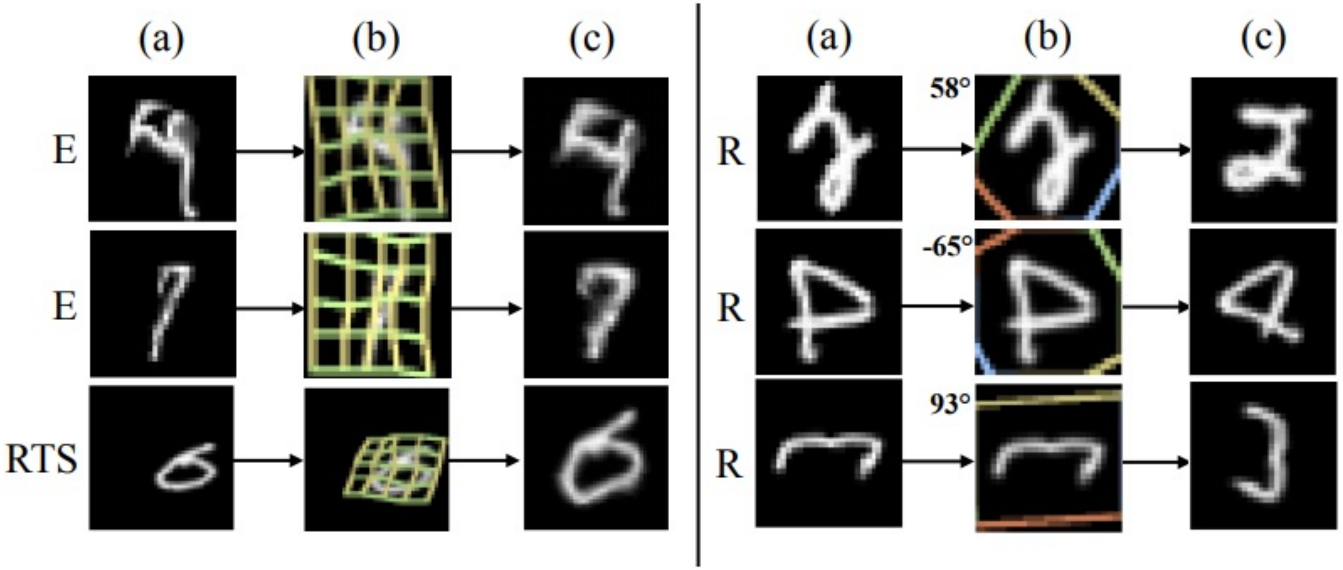
Sampler uses bilinear interpolation to produce output

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

Experiments



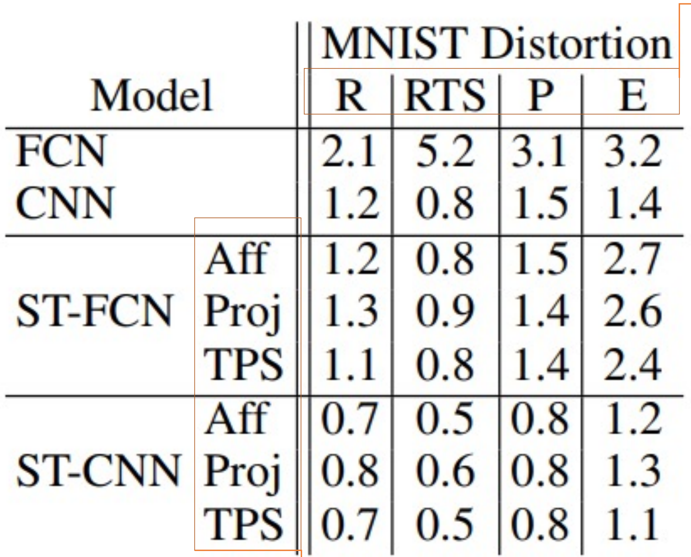
Model		MNIST Distortion			
		R	RTS	P	E
FCN		2.1	5.2	3.1	3.2
CNN		1.2	0.8	1.5	1.4
ST-FCN	Aff	1.2	0.8	1.5	2.7
	Proj	1.3	0.9	1.4	2.6
	TPS	1.1	0.8	1.4	2.4
ST-CNN	Aff	0.7	0.5	0.8	1.2
	Proj	0.8	0.6	0.8	1.3
	TPS	0.7	0.5	0.8	1.1



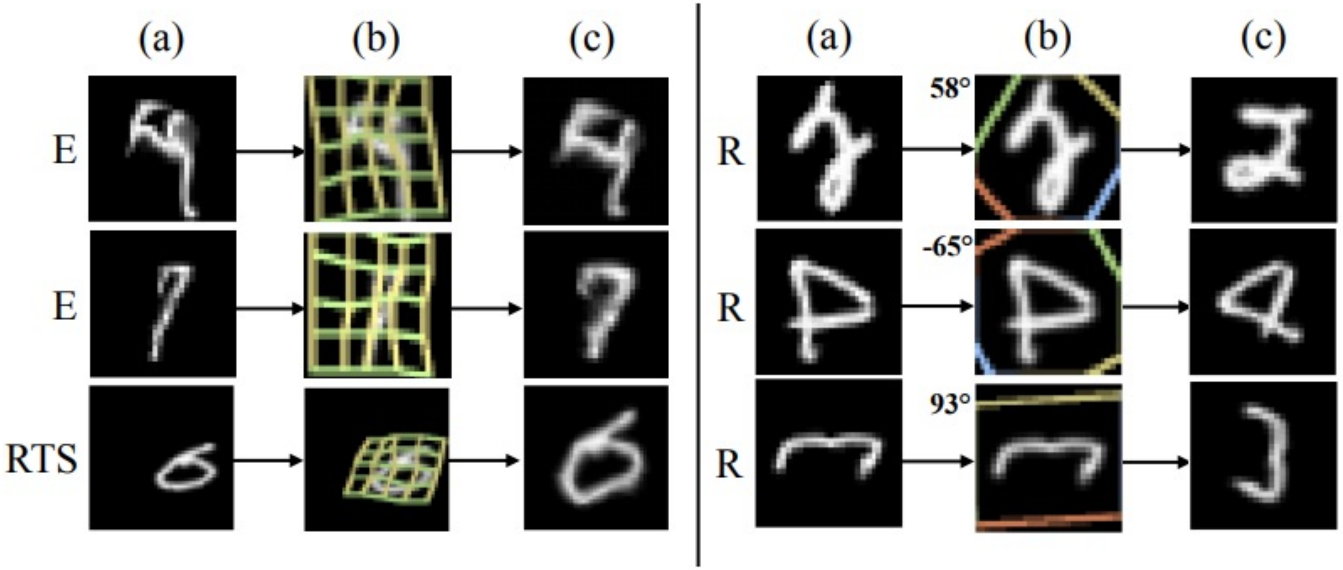
Experiments

Distortions:
R : rotated
RTS : rotated, translated, and scaled
P : projective distortion
E : elastic distortion.

Baseline

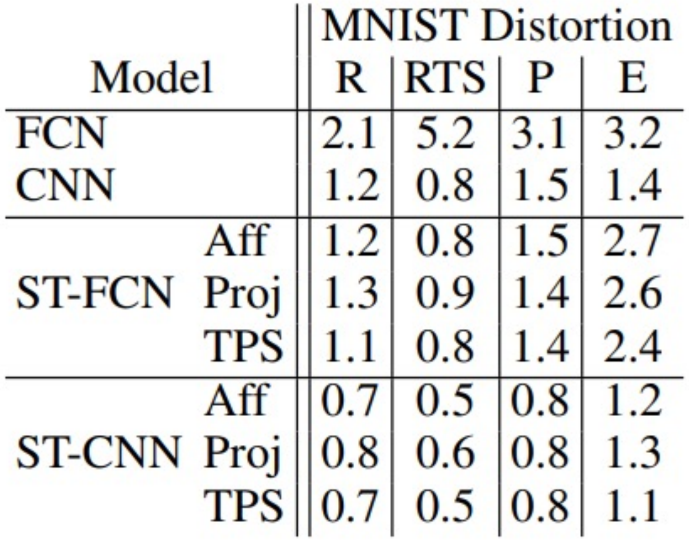


Model		MNIST Distortion			
		R	RTS	P	E
FCN		2.1	5.2	3.1	3.2
CNN		1.2	0.8	1.5	1.4
ST-FCN	Aff	1.2	0.8	1.5	2.7
	Proj	1.3	0.9	1.4	2.6
	TPS	1.1	0.8	1.4	2.4
ST-CNN	Aff	0.7	0.5	0.8	1.2
	Proj	0.8	0.6	0.8	1.3
	TPS	0.7	0.5	0.8	1.1

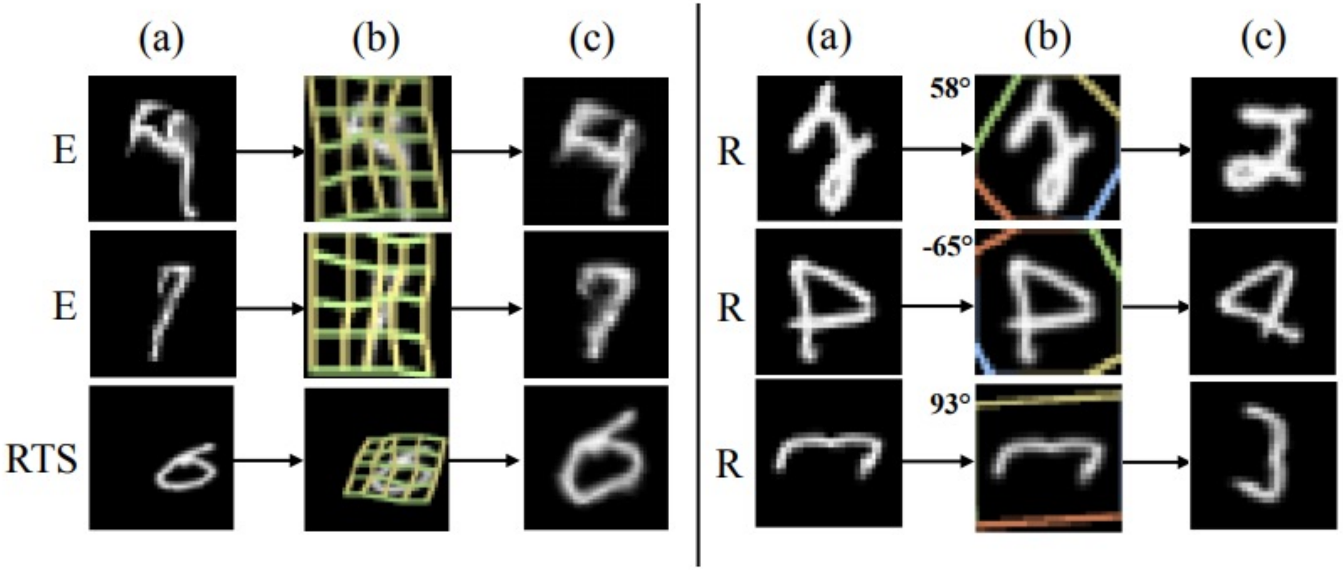


Transformations:
Aff : Affine
Proj : Projective
TPS : Thin Plate Spline

Experiments

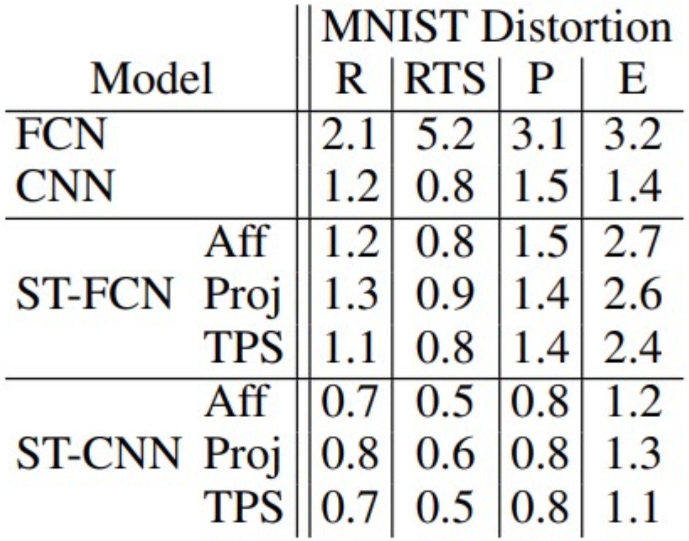


Model		MNIST Distortion			
		R	RTS	P	E
FCN		2.1	5.2	3.1	3.2
CNN		1.2	0.8	1.5	1.4
ST-FCN	Aff	1.2	0.8	1.5	2.7
	Proj	1.3	0.9	1.4	2.6
	TPS	1.1	0.8	1.4	2.4
ST-CNN	Aff	0.7	0.5	0.8	1.2
	Proj	0.8	0.6	0.8	1.3
	TPS	0.7	0.5	0.8	1.1

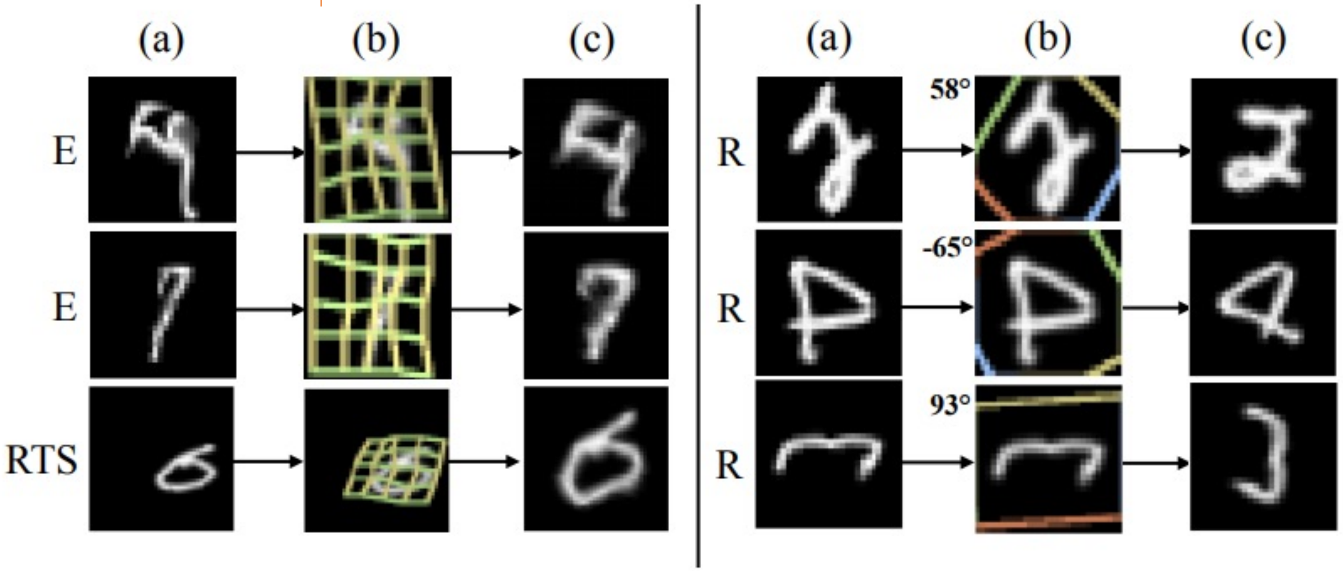


Experiments

a : input images.
b : transformations predicted by the spatial transformers.
c : outputs of the spatial transformers.



		MNIST Distortion			
Model		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



References

- Jaderberg, M. and Simonyan, K. and Zisserman, A. and kavukcuoglu, k., Spatial Transformer Networks, Advances in Neural Information Processing Systems, 2015, NIPS.

Sources

- https://cs231n.github.io/convolutional-networks/
- https://www.slideshare.net/xavigiro/spatial-transformer-networks
- https://medium.com/@manjunathbhat9920/spatial-transformer-network-82666f184299

THANK YOU.