Semi-supervised training of CNNs

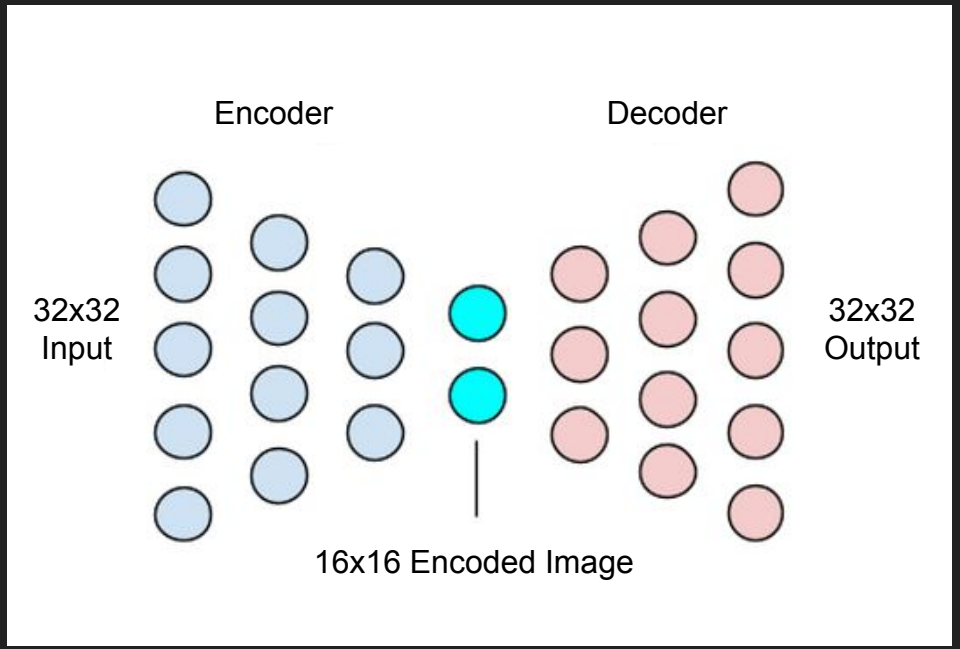
Student: Kevin Duarte
Mentors: Dr. Boqing Gong
Yang Zhang

Semi-supervised training of CNNs

- Semi-supervised learning typically makes use of a small amount of labeled data and a large amount of unlabeled data
  - Manually labeling images is a time consuming process, so reducing the amount of labeled data is ideal
- Through the use of unlabeled images, we can train a convolutional auto-encoder to compress and decompress an image
- We can initialize a classification CNN with the auto-encoder’s weights, and train it using a small set of labeled images
- The result is a CNN that can classify images with the accuracy of a similar CNN trained on a large set of labeled data

Results

- I have created an auto-encoder that compress and decompress a 32x32 image
- This auto-encoder was trained on 50,000 images and tested on 10,000 images
- It has an L2 loss of 1.12645



Results

- Using the auto-encoder's weights, we trained a CNN on 10,000 labeled images
- We also trained a randomly initialized CNN on the same 10,000 labeled images, as well as one trained on 50,000 labeled images
- The results show that the semi-supervised CNN had a similar accuracy to the CNN trained on 50,000 images, and much better accuracy than the randomly initialized CNN trained on 10,000 images

CNN	Accuracy
Semi-supervised, trained on 10,000 images	55.51%
Randomly initialized, trained on 10,000 images	48.47%
Randomly initialized, trained on 50,000 images	55.79%

Stacked What-Where Auto-encoder

- We have decided to create a stacked what-where auto-encoder (SWWAE) to replace the “naive” auto-encoder that I am currently using
  - The SWWAE has been shown to have extremely good results when used in semi-supervised training of CNNs
- Instead of using regular max-pooling and upsampling, the SWWAE saves spatial information at each max-pooling step, which is used in the unpooling
- The SWWAE also calculates the loss at intermediate reconstruction layers, instead of just at the final reconstruction of the input image

Stacked What-where Auto-encoder

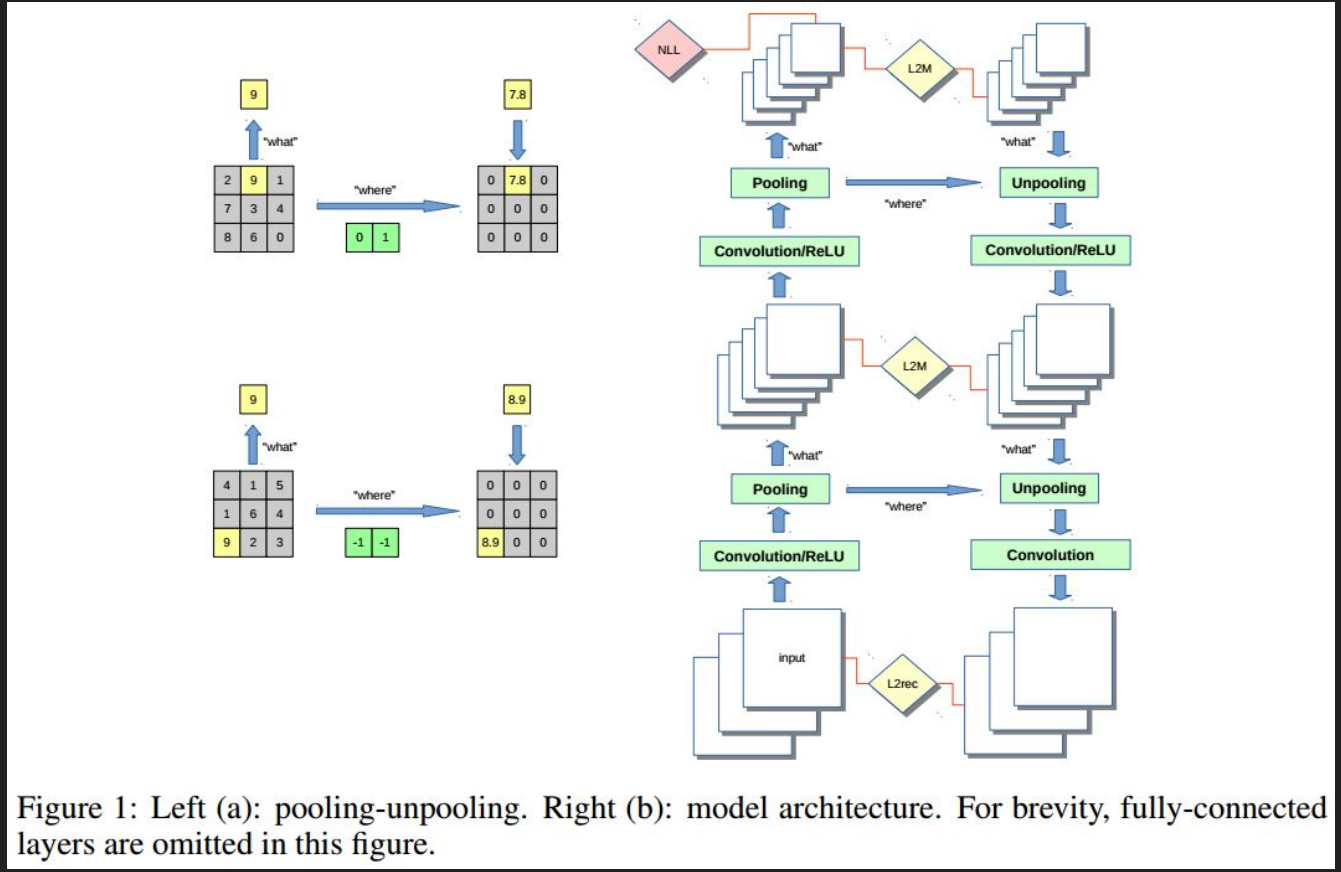


Figure 1: Left (a): pooling-unpooling. Right (b): model architecture. For brevity, fully-connected layers are omitted in this figure.

Stacked What-Where Auto-encoder Results

- I have successfully implemented the what-where auto-encoder's pooling and unpooling layers in theano/keras, as well as the intermediate reconstruction loss functions
- I am currently testing the pooling/unpooling and the intermediate loss functions on a shallow auto-encoder

Auto-encoder	Total Loss
Regular pooling/unpooling, only final reconstruction loss	1.54628
Regular pooling/unpooling, with intermediate reconstruction loss	2.04911
What-where pooling, only final reconstruction loss	0.58936
What-where pooling, with intermediate reconstruction loss	Currently training

Coming Weeks

- Once I am done training this shallow auto-encoder, I will use its weights to train a classification CNN
  - These results should be an improvement from the “naive” auto-encoder’s semi-supervised training results
- The final goal is to use a SWWAE for semi-supervised training of deep CNNs