# CS420 Machine Learning
# Project: on MNIST dataset

Mengyun Chen 716030210013
Yuge Zhang 716030210014

June 14, 2018

# Analysis

- Dataset technical specifications
  1. Training sample size: 60000.
  2. Features: $45 \times 45$.
  3. Color: Grey. Integer range in $[0, 255]$.
  4. Labels: Integer in $[0, 9]$.
  5. Test sample size: 10000.
- Difference from standard MNIST dataset
  1. The image is randomly extended with black blocks.
  2. Some random noise has been added to the image.

# Preprocessing

- Centering: $45 \times 45 \rightarrow 32 \times 32$.

- Thresholding: $[0, 255] \rightarrow \{0, 1\}$.
- Helpful? Hopeless?

# Some approaches everyone might try...

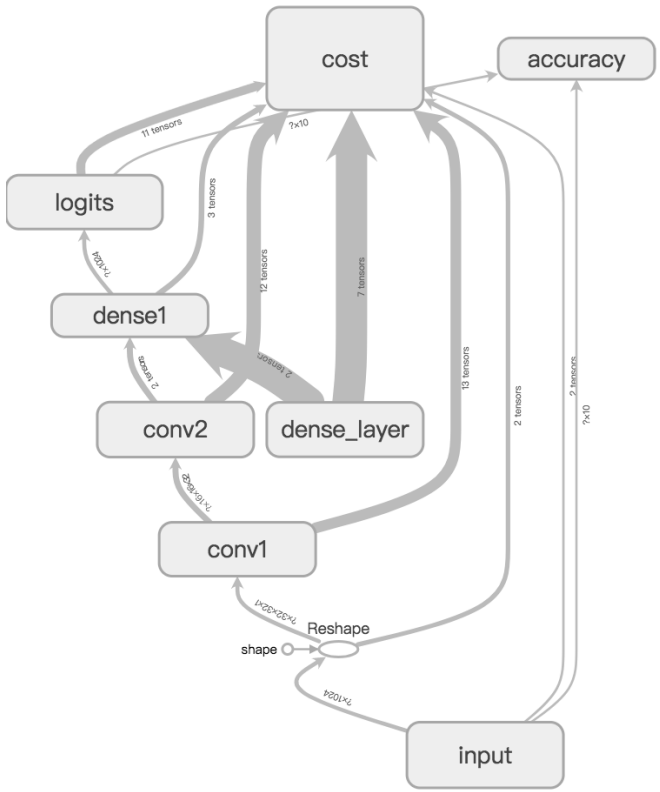
- $k$-Nearest Neighbors?
- Support Vector Machine?
- Simple Neural Network provided by sklearn?
- PCA?
  - Speedup?
  - Improve accuracy?

# Experiments – MLP, kNN, SVM

| Method | Preprocess | Accuracy | Time (second) |
|---|---|---|---|
| MLP | None | 0.6027 | 296.6 |
| MLP + PCA | None | 0.8372 | 100.6 |
| MLP + PCA | Centering | 0.9432 | 112.7 |
| kNN | Thresholding | 0.7057 | 368.5 |
| kNN | Centering | 0.9306 | 196.9 |
| kNN | Thres + Cent | 0.9011 | 161.3 |
| SVM + PCA | Thresholding | 0.882 | 353.3 |
| SVM + PCA | Centering | **0.9678** | 144.1 |
| SVM + PCA | Thres + Cent | 0.9641 | 147.4 |

# CNN: Another approach everyone might try (1/3)

A reimplemented version with Tensorflow low-level API. No dropout.

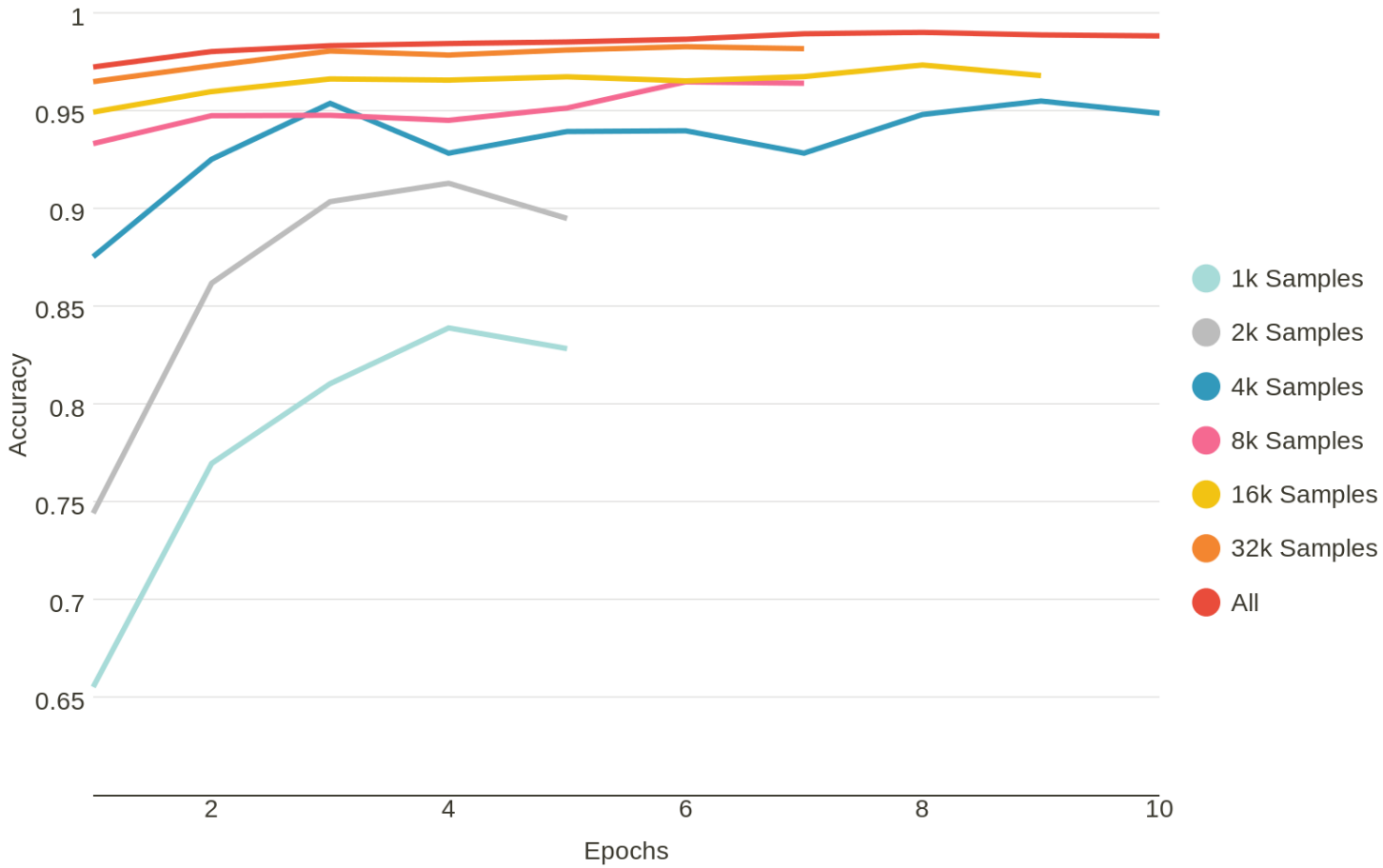


# CNN: Another approach everyone might try (2/3)

| Epoch | Loss | Time | Accuracy |
|---|---|---|---|
| 1 | 0.196677 | 57.9 | 0.9723 |
| 2 | 0.055729 | 60.2 | 0.9802 |
| 3 | 0.028606 | 60.7 | 0.9832 |
| 4 | 0.018365 | 61.4 | 0.9843 |
| 5 | 0.015604 | 62.2 | 0.9851 |
| 6 | 0.012914 | 60.5 | 0.9865 |
| 18 | 0.005556 | 61.1 | 0.9905 |
| 19 | 0.004799 | 61.5 | 0.9896 |
| 20 | 0.00405 | 61.7 | 0.9905 |

# CNN: Another approach everyone might try (3/3)

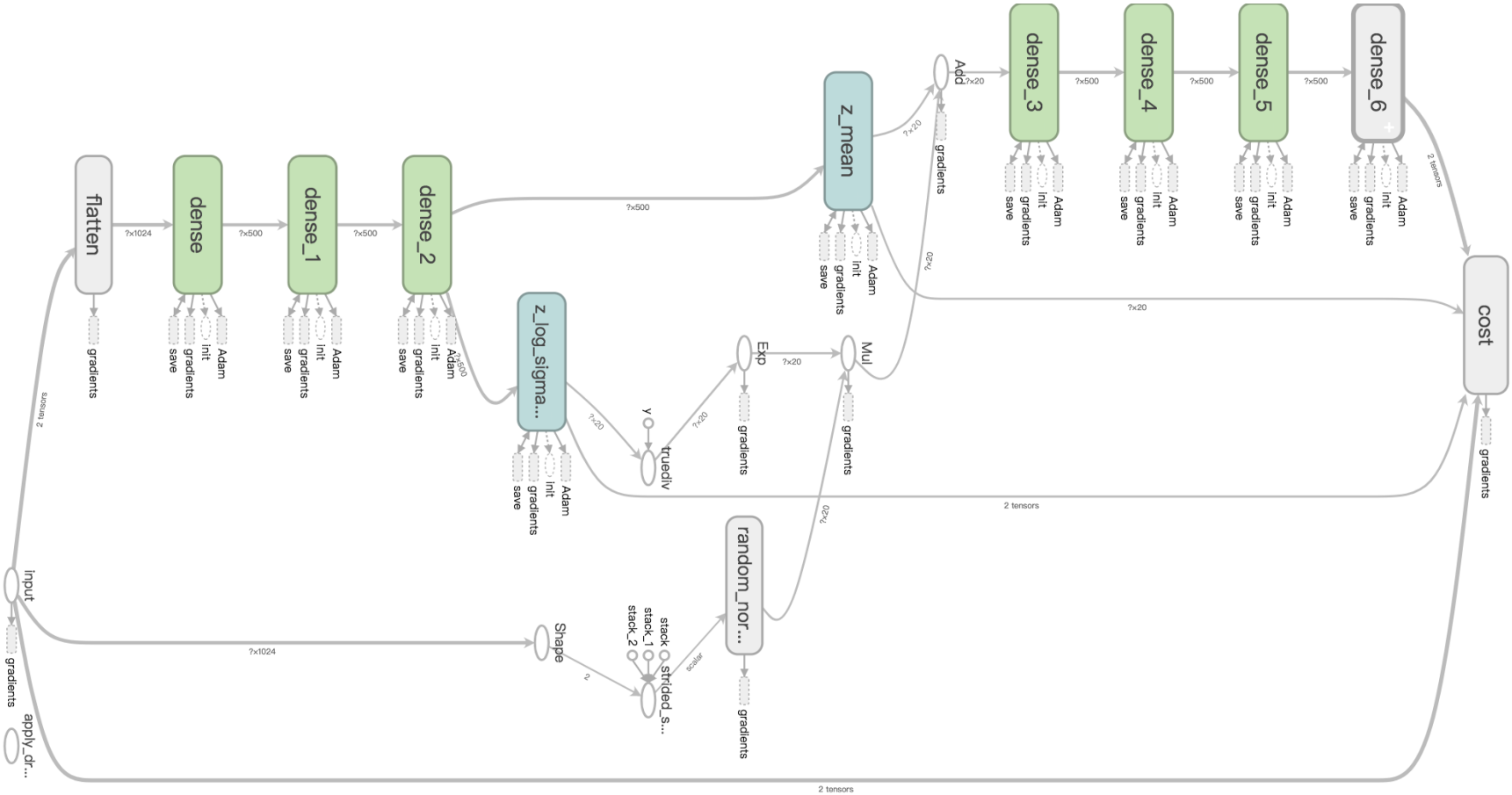
Performance depends on training sample size?



# Variational Autoencoder (VAE)

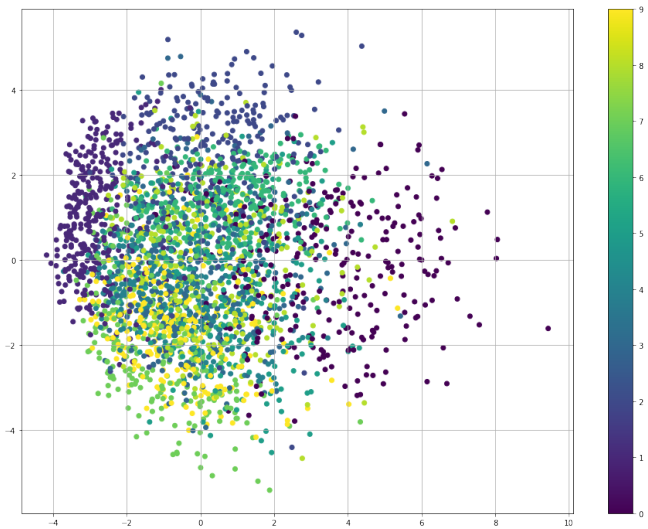
1. Unsupervised. So... bad idea?
2. Replace PCA.
3. Denoise.
4. Fancy visualization.

# VAE Network Architecture

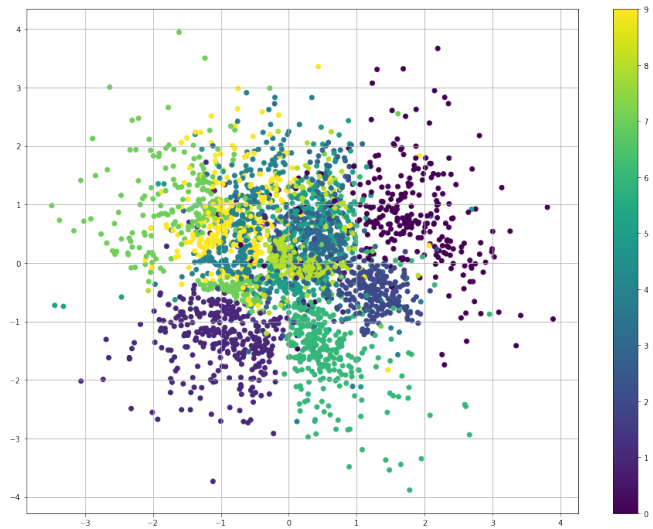


# VAE: Dimensionality reduction

Dimension of latent space = 2.

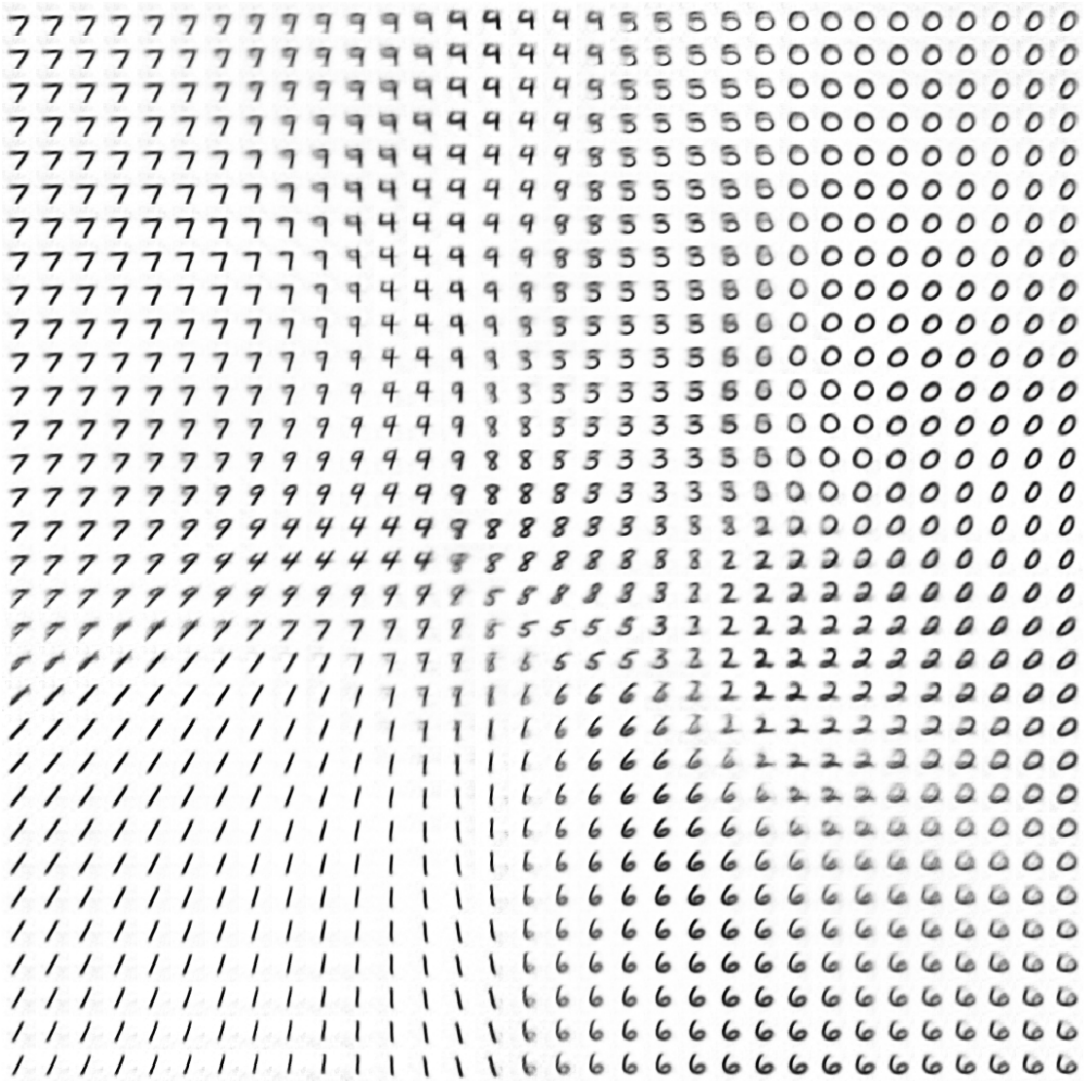


(a) Result with PCA



(b) Result with VAE

# VAE: Reconstruction



# VAE: With CNN

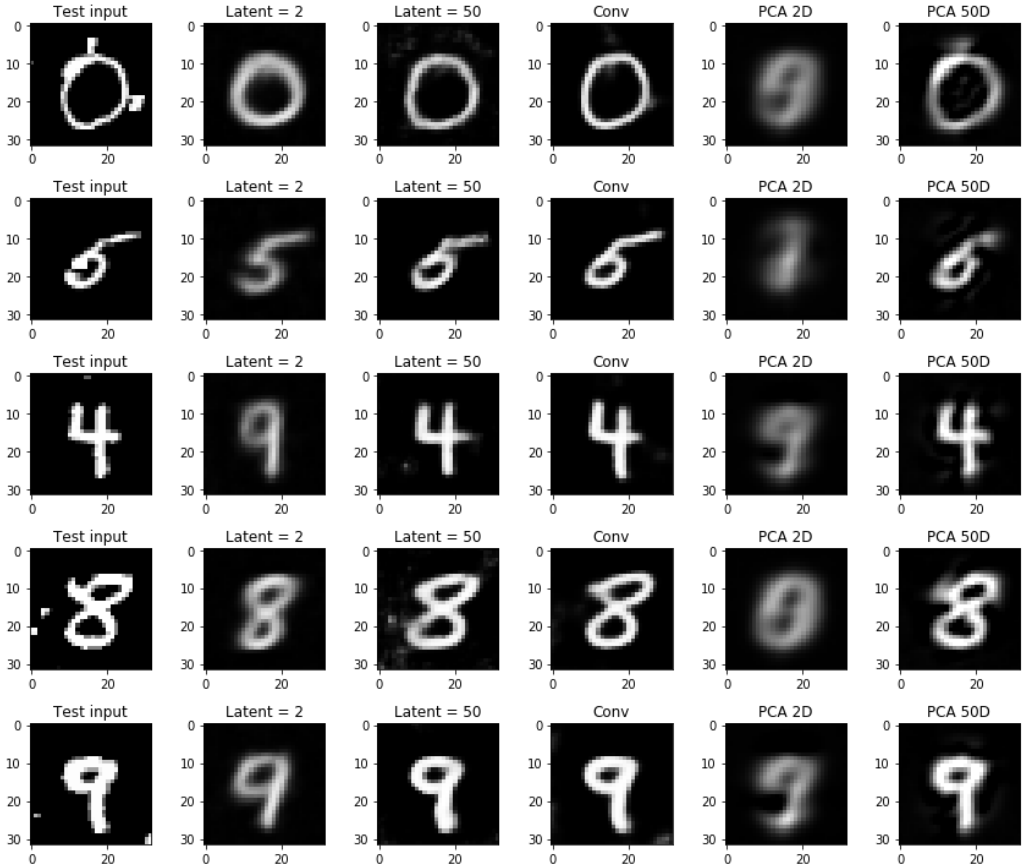



Figure: VAE 2D / VAE 50D / VAE CNN / PCA 2D / PCA 50D

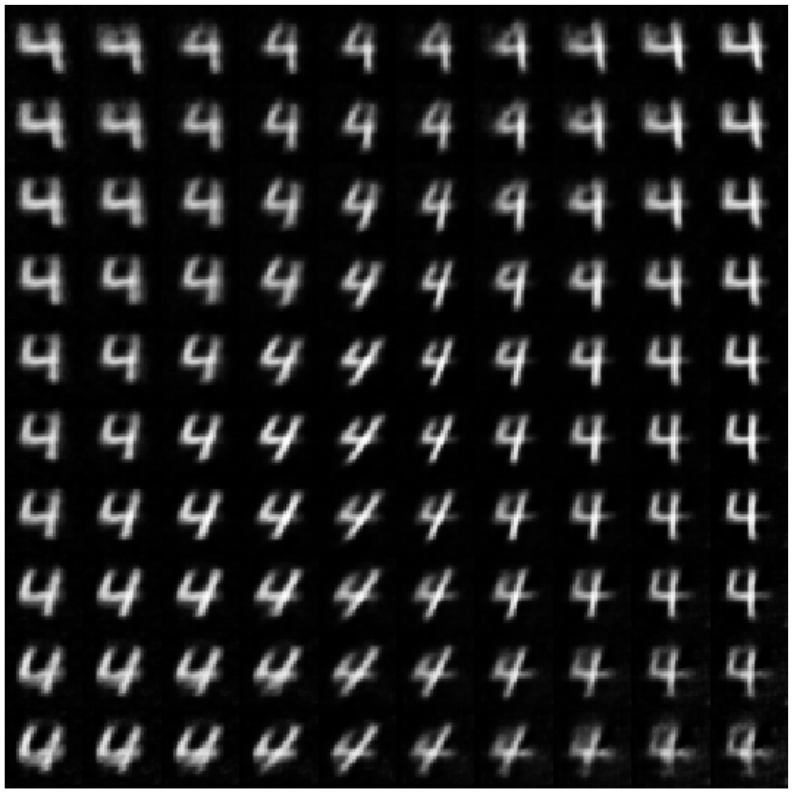
# VAE: Helps classification

- Work as PCA.
- Use SVM in the following step.

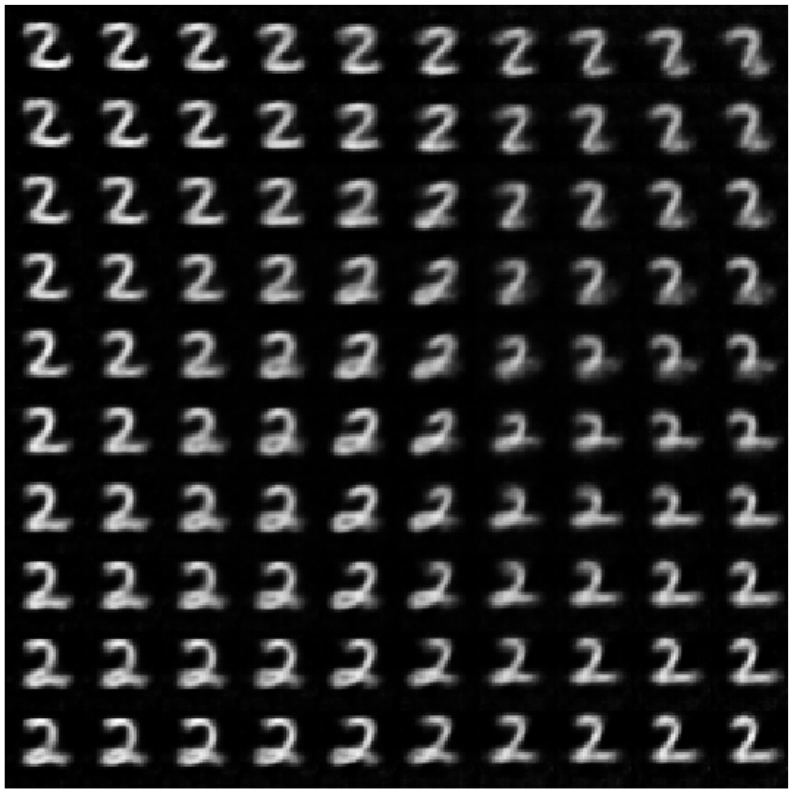
| Method | Preprocess | Accuracy | Time |
|---|---|---|---|
| VAE + SVM (latent dim = 2) | Centering | 0.6944 | $\approx$ 7 mins |
| VAE + SVM (latent dim = 50) | Centering | 0.9403 | $\approx$ 10 mins |
| VAE (Conv) + SVM | Centering | **0.9634** | $\approx$ 8 hours |

# Conditional Variational Auto Encoder

Control the label and generate different styles of a number.



(a) Different styles of 4



(b) Different styles of 2

# CVAE: Classification (1/4)

**Algorithm 1** $\textsc{Predict}(x)$

**Require:** $\boldsymbol{x}$ is a $n$-dimension vector.

**for all** $y$ in $[0, 9]$ **and** $y$ is an integer **do**

regenerate $\boldsymbol{x'}$ with $(\boldsymbol{x}, y)$.

$\textsc{BestLoss} \leftarrow \infty$

**if** $\textsc{ReconstrLoss}(\boldsymbol{x}, \boldsymbol{x'}) < \textsc{BestLoss}$ **then**

$\textsc{BestLoss} \leftarrow \textsc{ReconstrLoss}(\boldsymbol{x}, \boldsymbol{x'})$

$\textsc{Label} \leftarrow y$

**end if**

**end for**

**return** $\textsc{Label}$

# CVAE: Classification (2/4)

- $\textsc{ReconstrLoss} = \boldsymbol{x} \log(\boldsymbol{x'}) + (1 - \boldsymbol{x}) \log(1 - \boldsymbol{x'})$.
- Hit accuracy **0.9423** with 3 layers, 500 hidden units each, 20-dimension latent space and 15-minute training.
- Disappointing?

# CVAE: Classification (3/4)

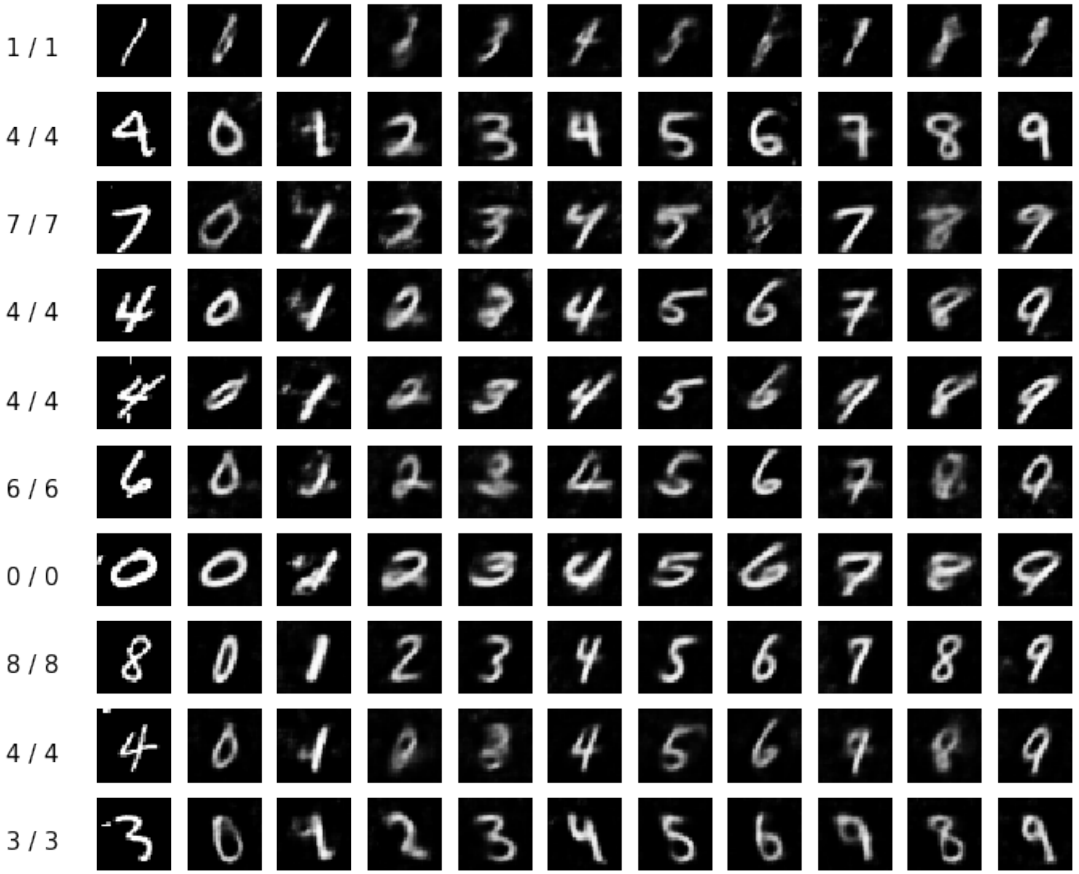
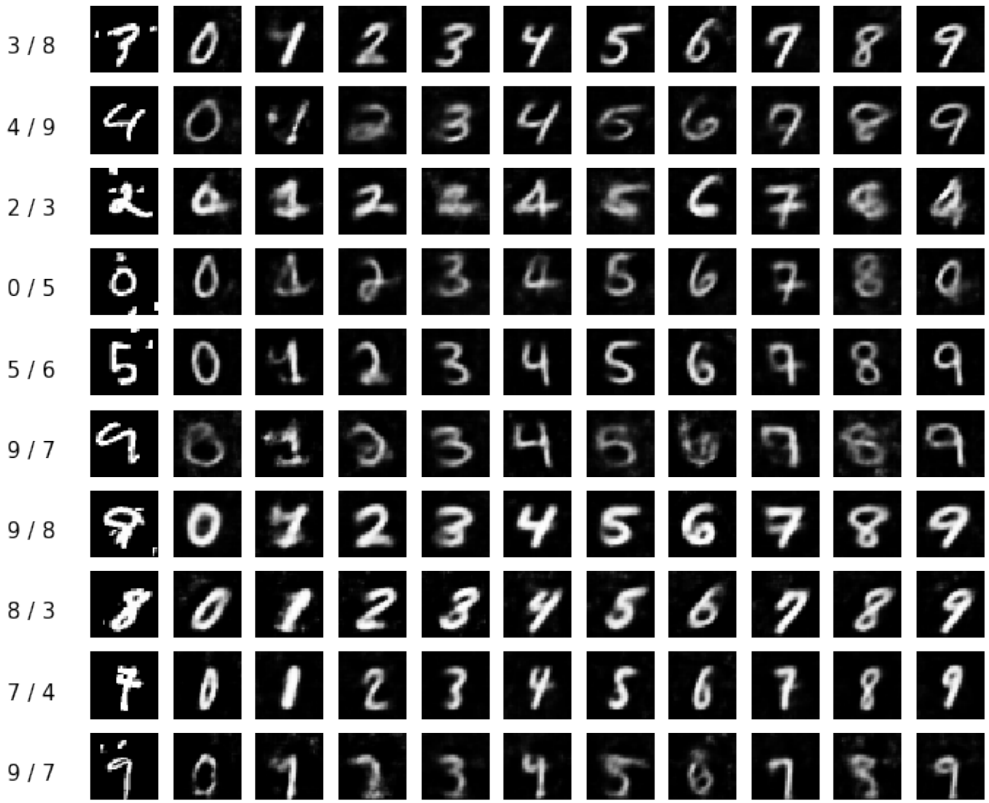



Figure: CVAE reconstruction with label from 0 to 9

# CVAE: Classification (4/4)




*CVAE: It doesn't look like anything to me...*

# Something else?

- Lack of fundings, we didn't use any GPU. All the tests on done on a server with Intel Xeon CPU E5-2650 v4 @ 2.20GHz with 24 cores and 32 GB RAM.
- The math behind VAE is responsible for a lot of headaches. Luckily, programming is easier than math...
- The project was designed to be simple, straight-forward, clichéd. It turned out to be way much more time-consuming and fascinating than expected.