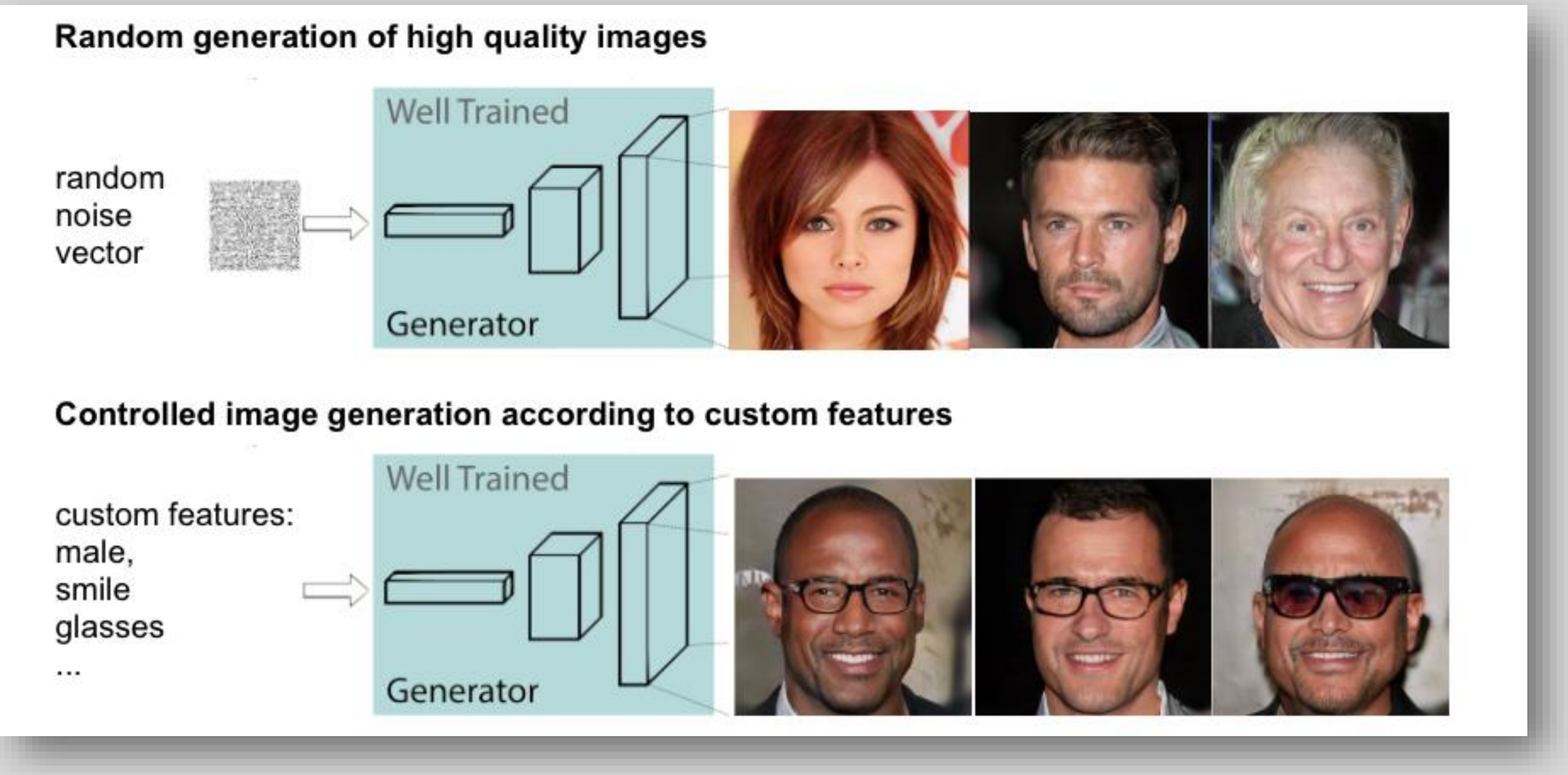
INTELLIGENT SYSTEMS FINAL PROJECT

MR Tavakoli & MM Alemohammad & M Heydari & MH Vaeedi



Random generation of high quality images

random noise vector

Well Trained

Generator

Controlled image generation according to custom features

custom features:
male,
smile
glasses
...

Well Trained

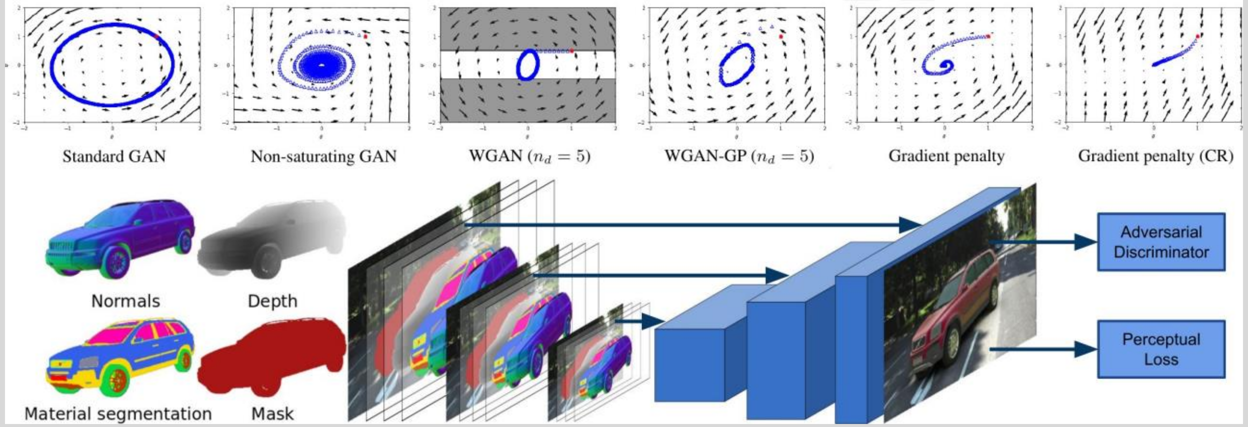
Generator

All these persons are fake!

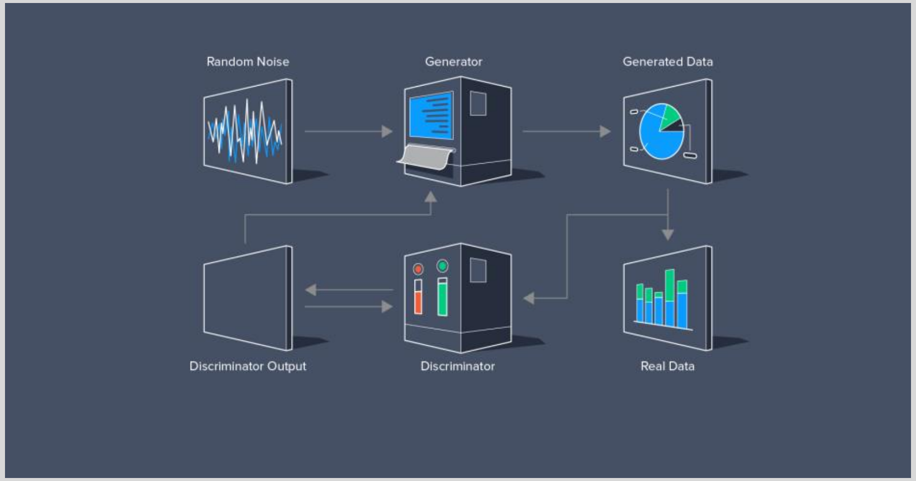
PART1.IMAGE GENERATION

Generative models!?

- Generative modeling is an unsupervised learning task .
- it involves automatically discovering and learning the regularities or patterns in input data
- The model can be used to generate or output new examples that plausibly could have been drawn from the original dataset.
- Real world applications: Deepfakes , face aging , 3D object generation , …



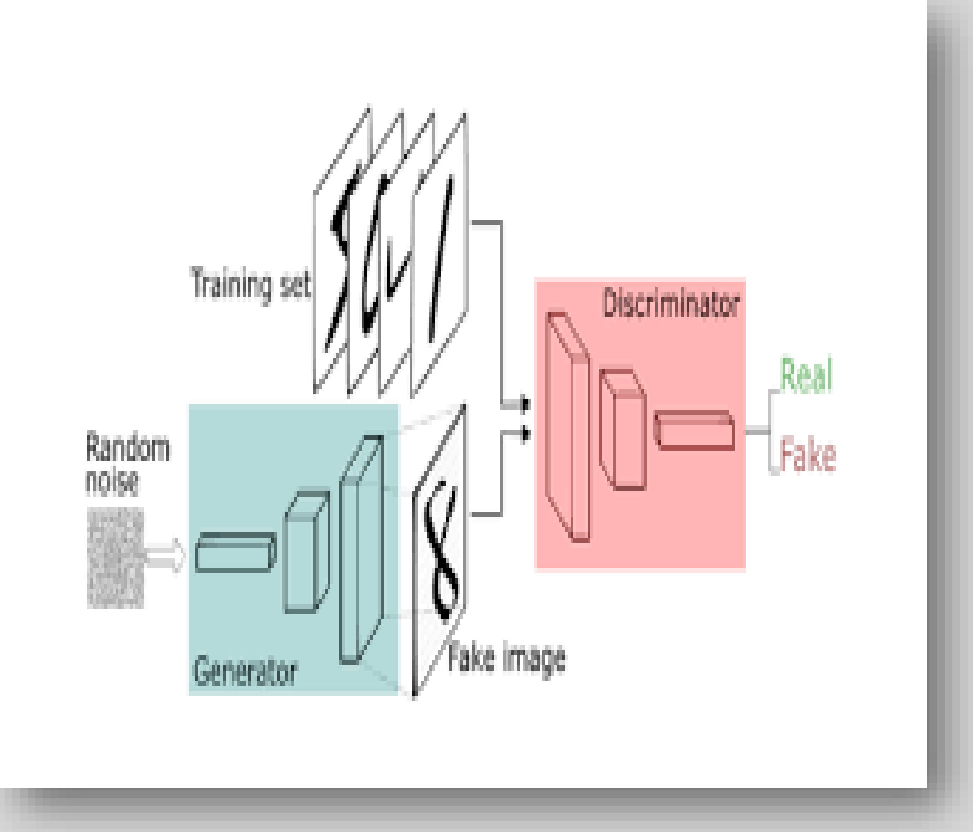
Generative Adversarial Networks



- Next step in generative models
- Consists of discriminative and generator models
- These 2 models are trained in competition with each other
- Discriminator : tells if image is fake or real
- Generator : generates new fake images using random noise vectors and deconvoluion layers

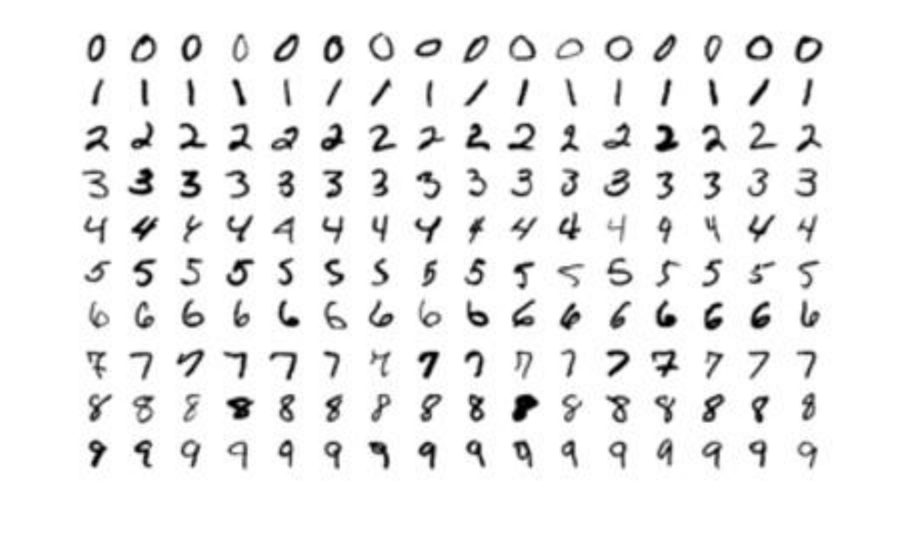
GAN at a glance

- Step 1: Define the problem.
- Step 2: Define architecture of GAN.
- Step 3: Train Discriminator on real data for n epochs.
- Step 4: Generate fake inputs for generator and train discriminator on fake data.
- Step 5: Train generator with the output of discriminator.



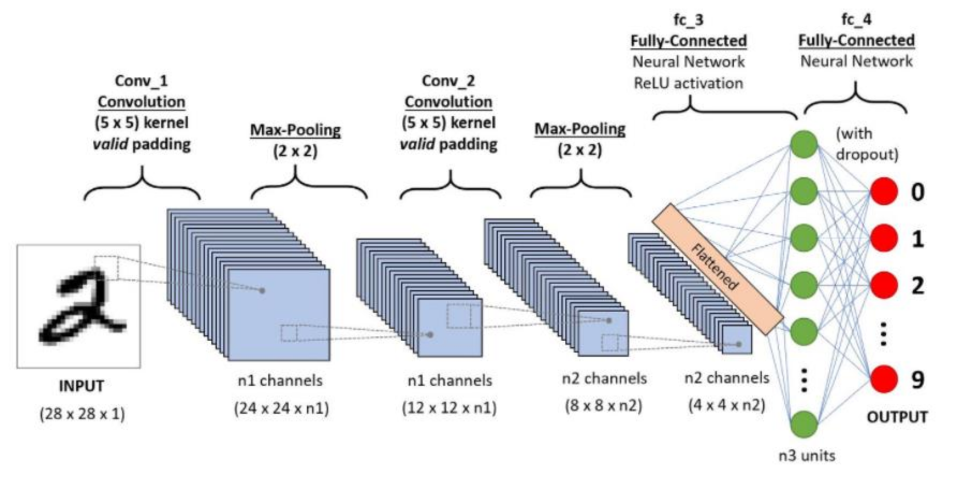
But , what we did?

- Using MNIST digits dataset
- Start with training a CNN classifier
- Provide random noise picture as input
- Back propagate to the very beginning and update the input using ADAM solver



CNN design

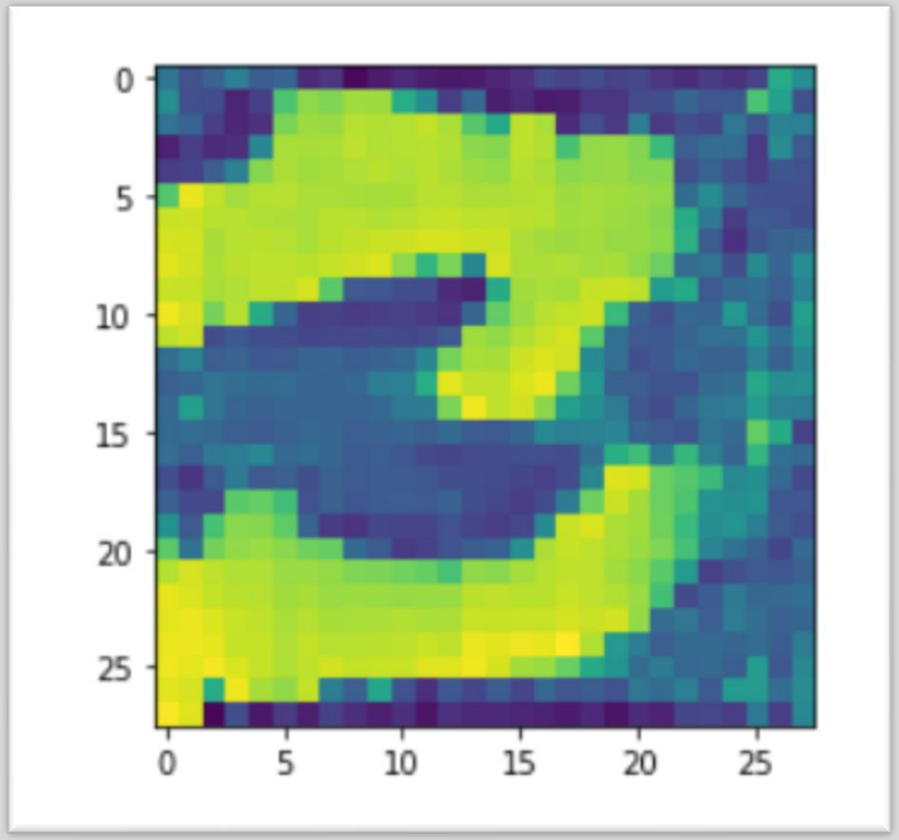
- 2 conv2d with (5*5*8) kernel size followed by (2*2*8) max pooling
- Flatten layer
- Fully connected layer with 200 neurons
- Relu activation functions
- Softmax for one-hot representation

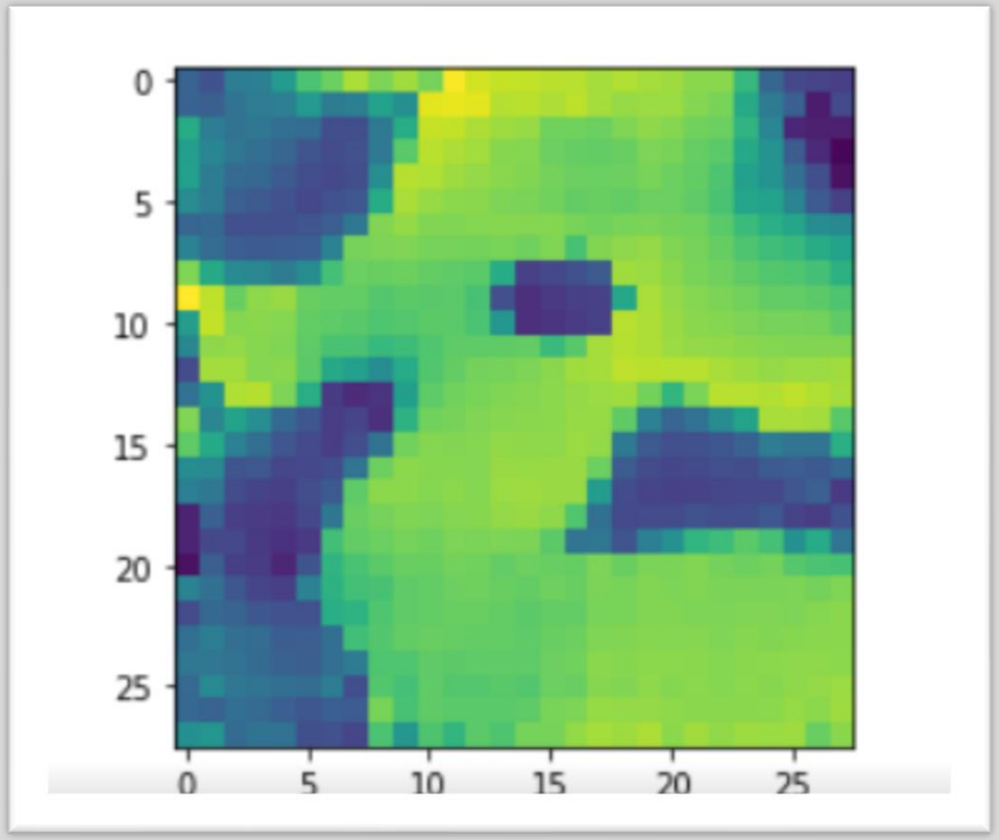


Classifier Results

- Learning rate = 0.001
- Beta1 = 0.9
- Beta2 = 0.999
- 4 epochs training on 5000 data points
- Achieving 91% accuracy

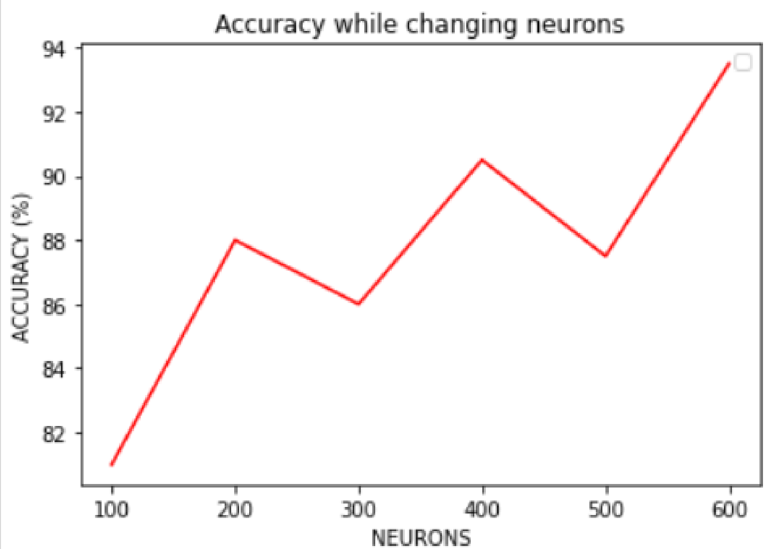
Generator results





Effect of changing neurons count

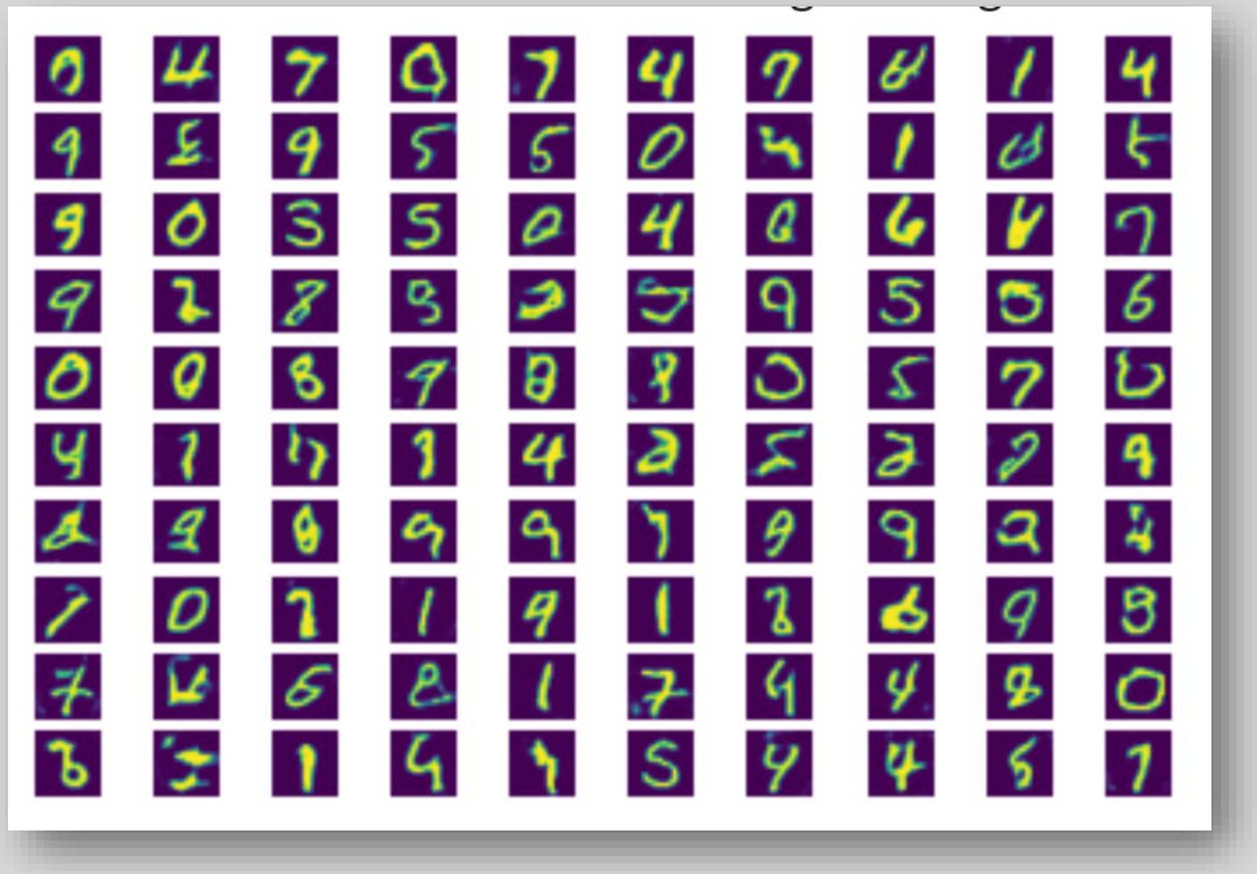
- 100 → 81%
- 200 → 88%
- 300 → 86%
- 400 → 90.5%
- 500 → 87.5%
- 600 → 93.5%
- A hyper parameter
- There is no common pattern here so testing to get results is required!



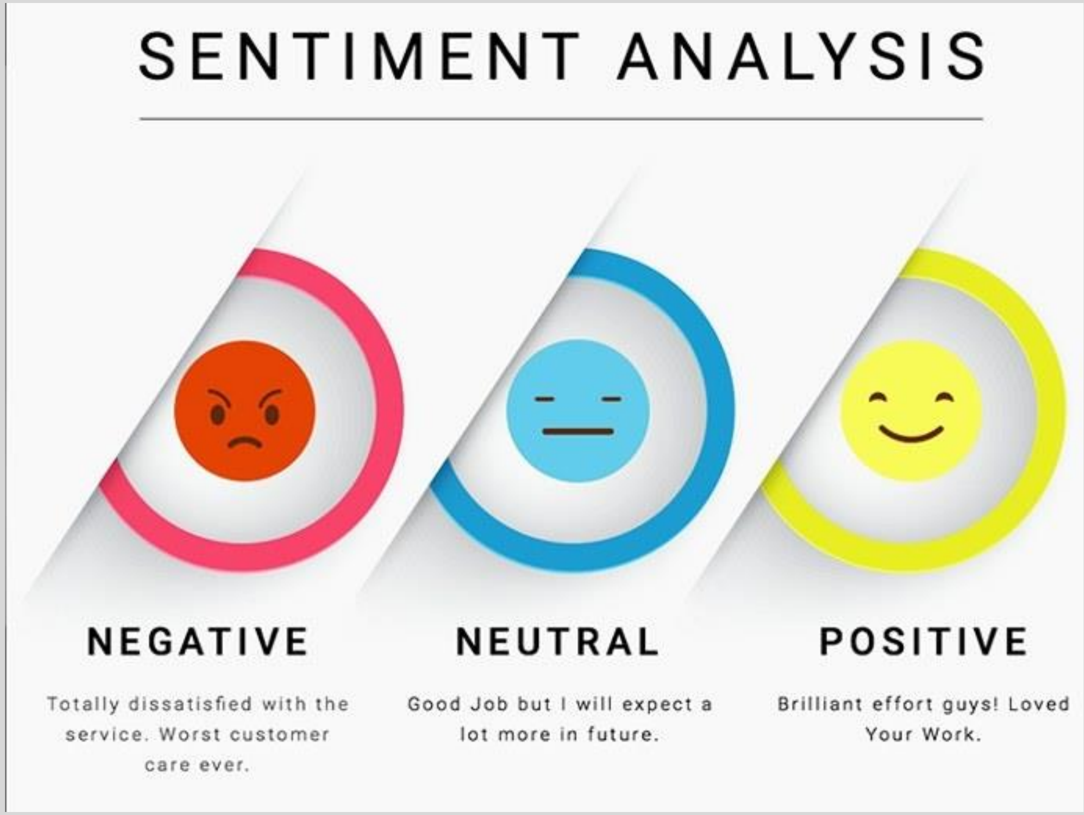
Effect of changing layers count

- 3 layer → 12%
- Dramatic decrease in accuracy indicates that change in layers count is not recommended.
- Unlike our common belief that increasing layers results in better accuracy its not the case here.

Improvements : GAN network



PART2, SENTIMENTAL ANALYSIS OF TEXT



SENTIMENT ANALYSIS

NEGATIVE

Totally dissatisfied with the service. Worst customer care ever.

NEUTRAL

Good Job but I will expect a lot more in future.

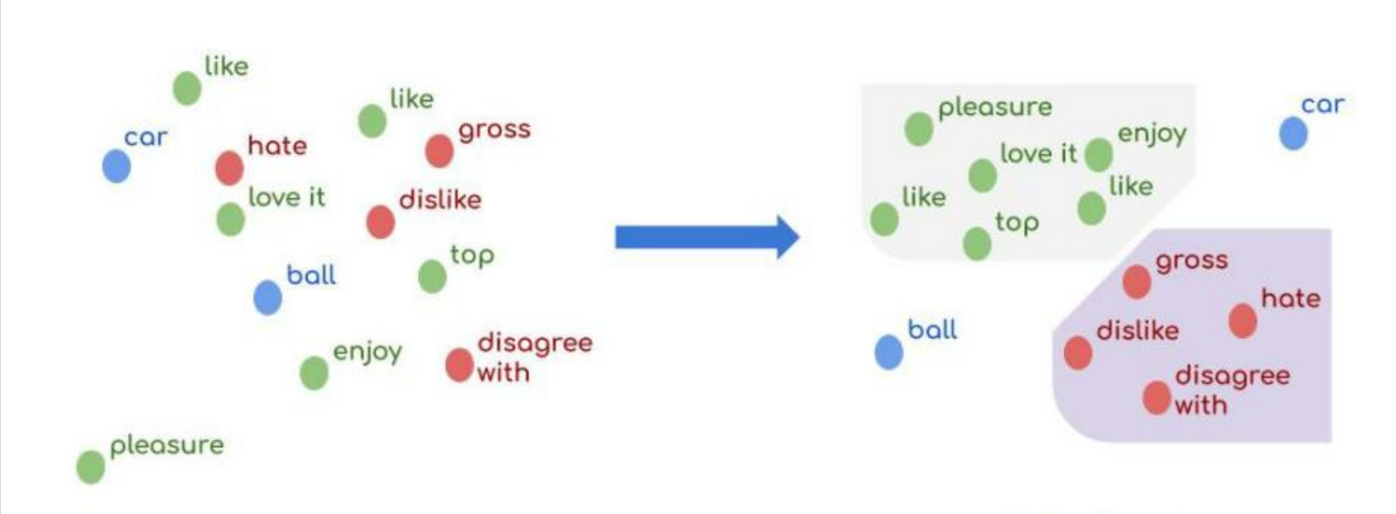
POSITIVE

Brilliant effort guys! Loved Your Work.

Computers can feel

Natural Language Processing

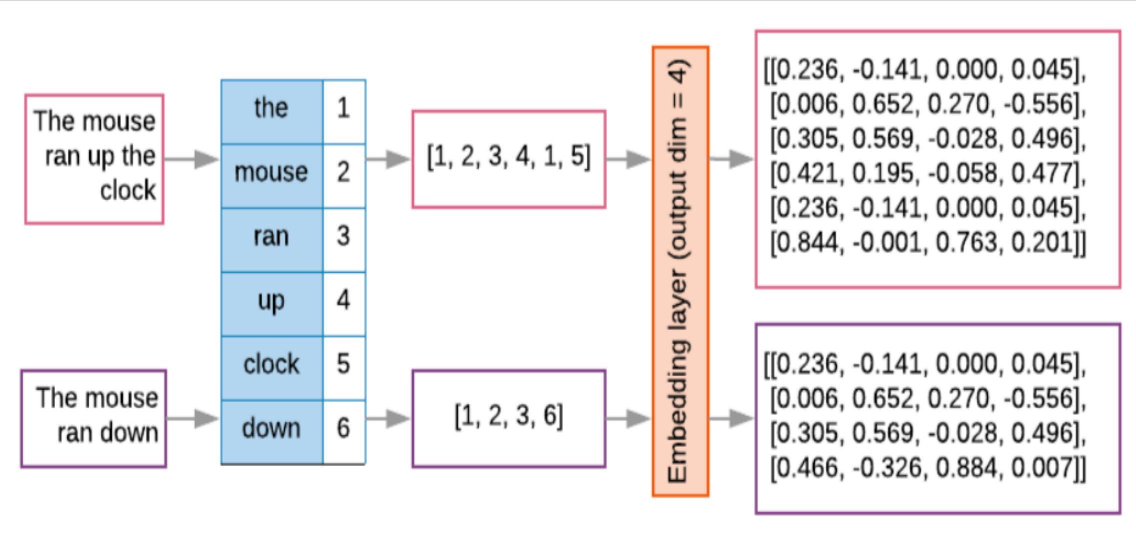
- Using pre-trained words to learn from texts
- GloVe, an unsupervised learning algorithm provided by Stanford
- Define each world with a n-element vector of numbers in n-dimensional space
- Embedded dictionary is a bridge between natural language and computer language



Sentence coding

- Reading feedbacks with labels
- Splitting each sentence to words
- Punctuations act as words
- Sentence is converted to a 3D matrix

  $(length\ of\ sentence \times 1 \times dimension\ of\ space)$
- We used 300-d space
- How to solve it ?



The mouse ran up the clock

The mouse ran down

the	1
mouse	2
ran	3
up	4
clock	5
down	6

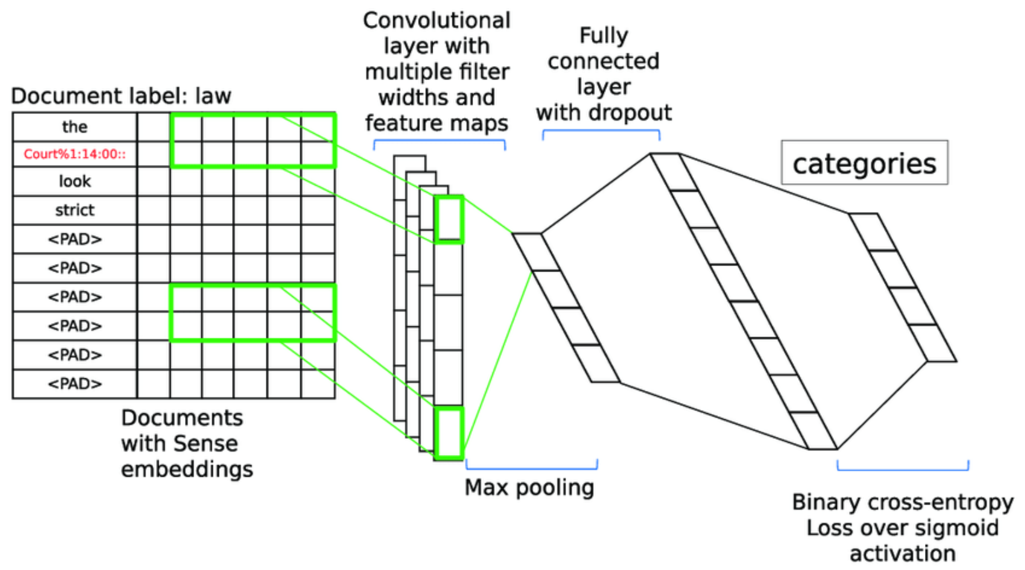
[1, 2, 3, 4, 1, 5]

[1, 2, 3, 6]

```
[[0.236, -0.141, 0.000, 0.045],
 [0.006, 0.652, 0.270, -0.556],
 [0.305, 0.569, -0.028, 0.496],
 [0.421, 0.195, -0.058, 0.477],
 [0.236, -0.141, 0.000, 0.045],
 [0.844, -0.001, 0.763, 0.201]]
```

```
[[0.236, -0.141, 0.000, 0.045],
 [0.006, 0.652, 0.270, -0.556],
 [0.305, 0.569, -0.028, 0.496],
 [0.466, -0.326, 0.884, 0.007]]
```

CNN as always ☺


Document label: law
the
Court%1:14:00::
look
strict
<PAD>
<PAD>
<PAD>
<PAD>
<PAD>
<PAD>
Documents with Sense embeddings
Convolutional layer with multiple filter widths and feature maps
Max pooling
Fully connected layer with dropout
categories
Binary cross-entropy Loss over sigmoid activation

- Matrix of each sentence is as same as the matrix of an image
- CNN cares about word place in sentence
- Long sentences were truncated
- Short sentences should be padded or wrapped
- Normalization plays an important role

Parameters and Network architecture

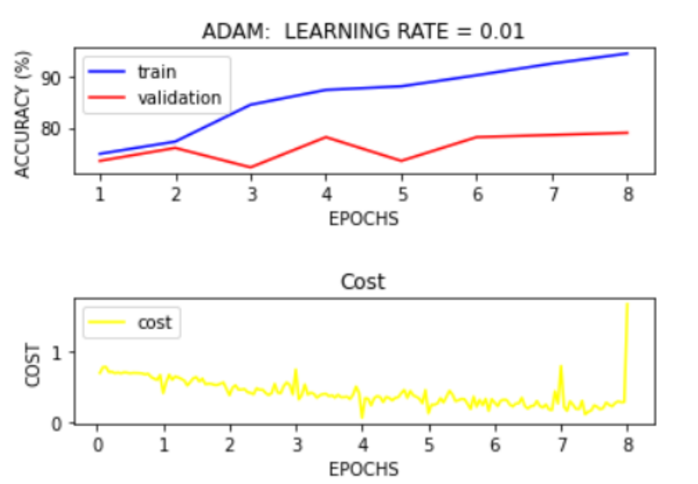
- ADAM solver
- Learning rate = 0.001
- Beta1 = 0.9, Beta2 = 0.999
- Batch size = 100
- Epochs = 8

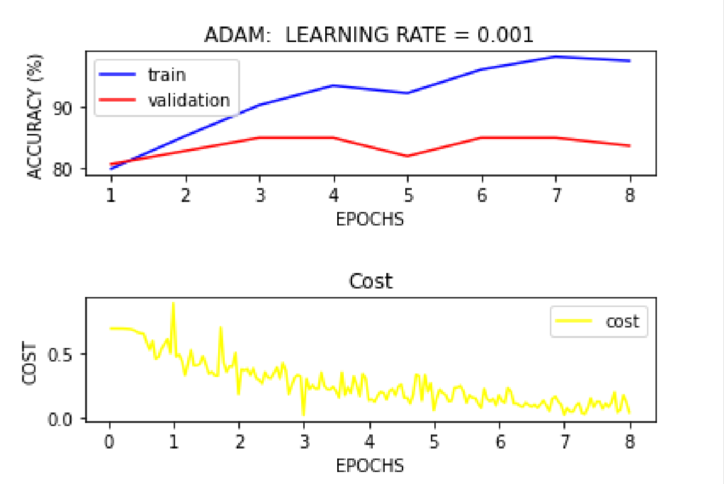
- 2 × conv2D with 64×(3, 1, 300) kernel
- Max pool, size (2, 1)
- Flatten
- Fully connected, 128 neurons
- Relu activation function
- Softmax one-hot representation

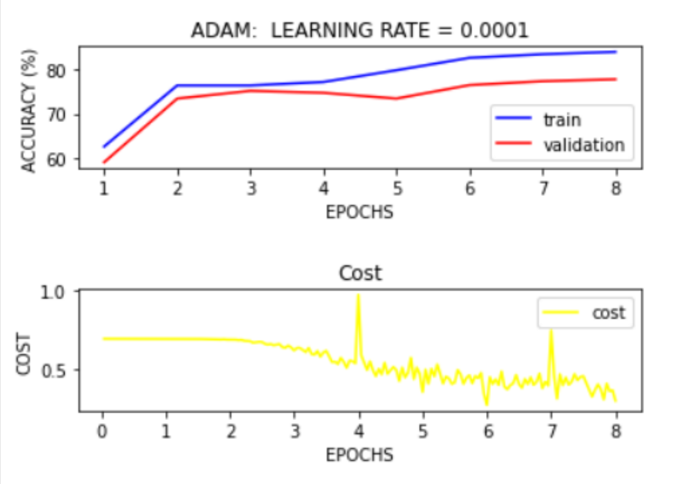
Changing learning rate (1)

Learning rate	Test accuracy
0.1	Did not converge
0.01	78.16 %
0.001	85.19 %
0.0001	83.25 %

Changing learning rate (2)



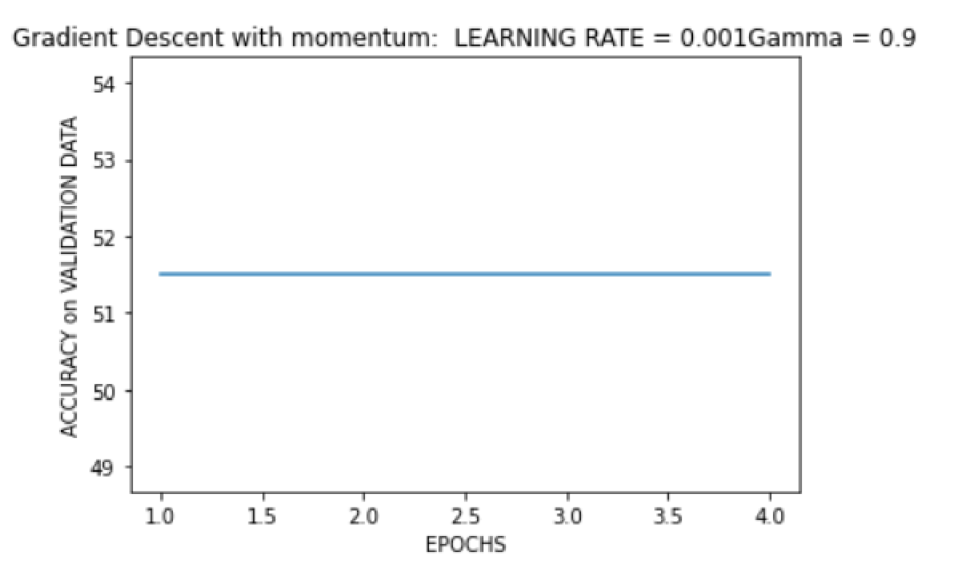


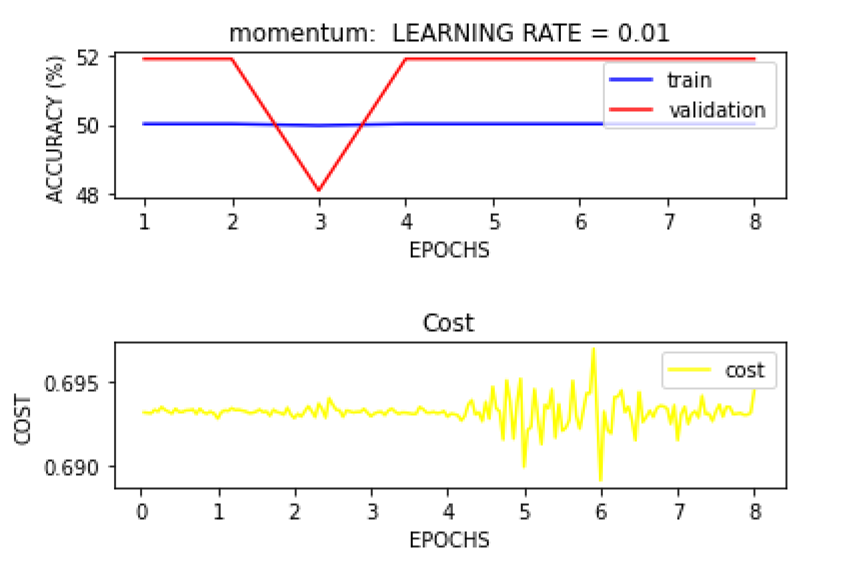


Different solver

- MOMENTUM (LR = 0.001)
- Test accuracy = 48.54 %

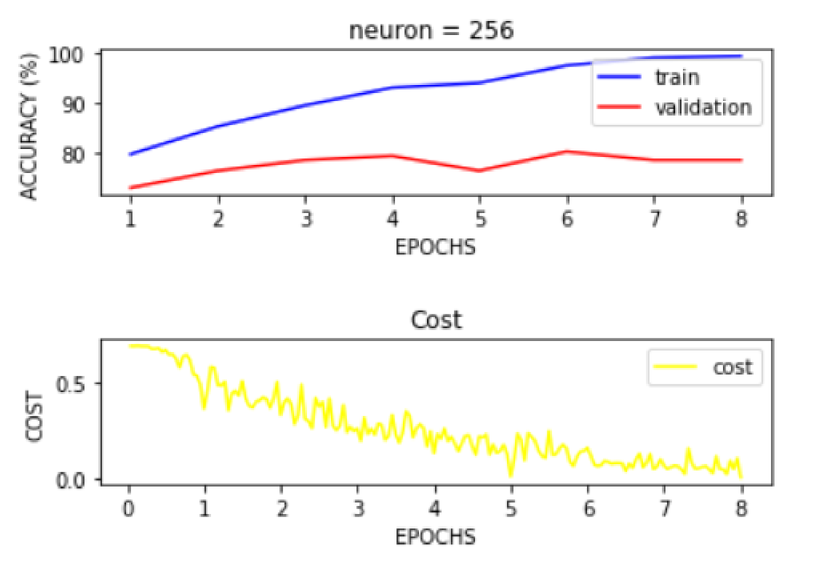
- MOMENTUM (LR = 0.01)
- Test accuracy = 51.7 %



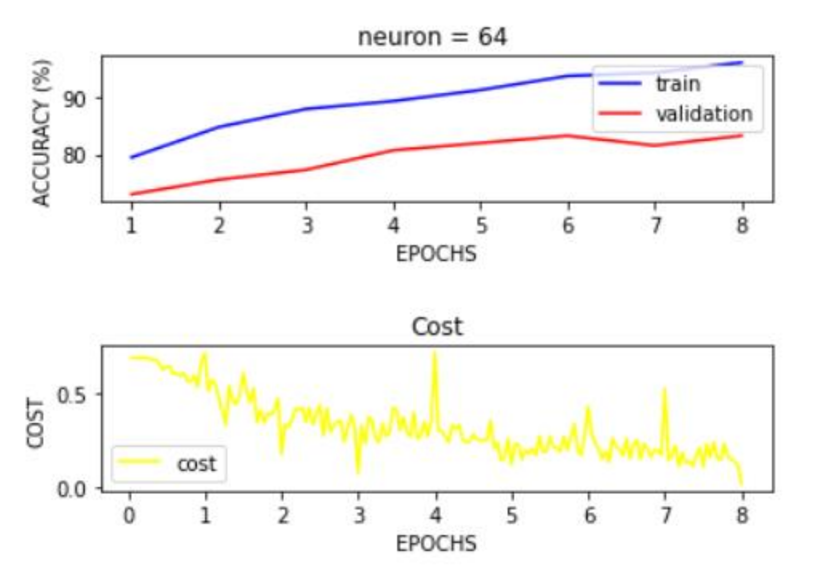


Changing number of neurons

- 128 → 256
- Test accuracy = 85.68 %

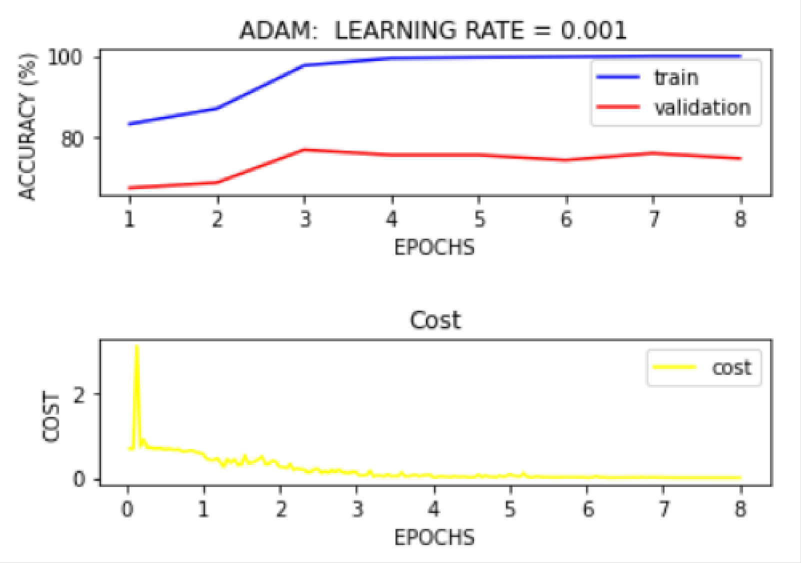


- 128 → 64
- Test accuracy = 85.19 %



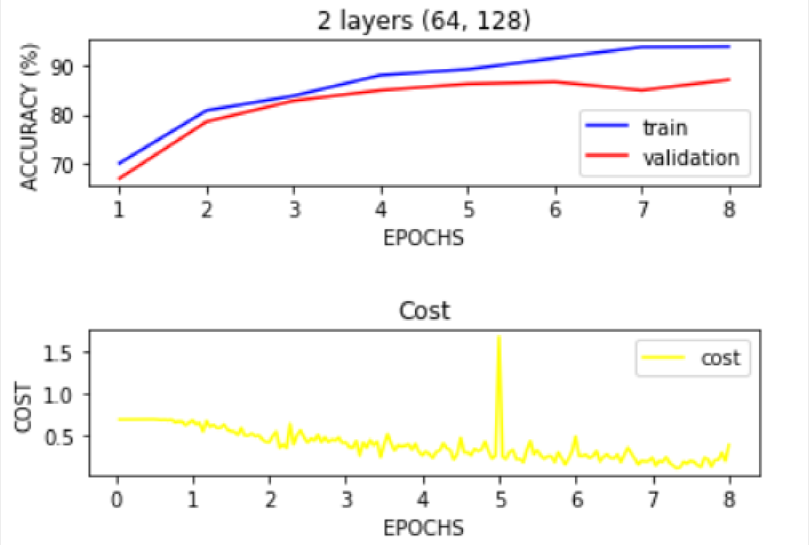
What about MLP !?

- First flatten the input data to a 15000 elements 1D array
- And just using 3 fully connected layers with size 1024
- <u>Results:</u>
  - Test accuracy = 73.79 %



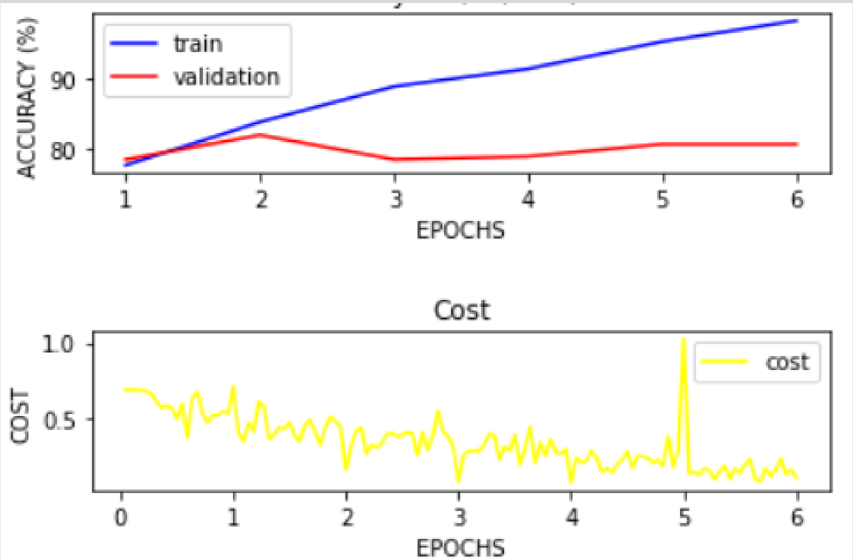
Changing number of layers

- Add another convolutional layer
- 2 × conv2D with 128×(3, 1, 300) kernel
- Max pool, size (2, 1)
- Relu activation function



Linear activation function

- Change Relu to $y = 1x + 0$
- Results:
  - Test accuracy = 48.82 %



Thank you!