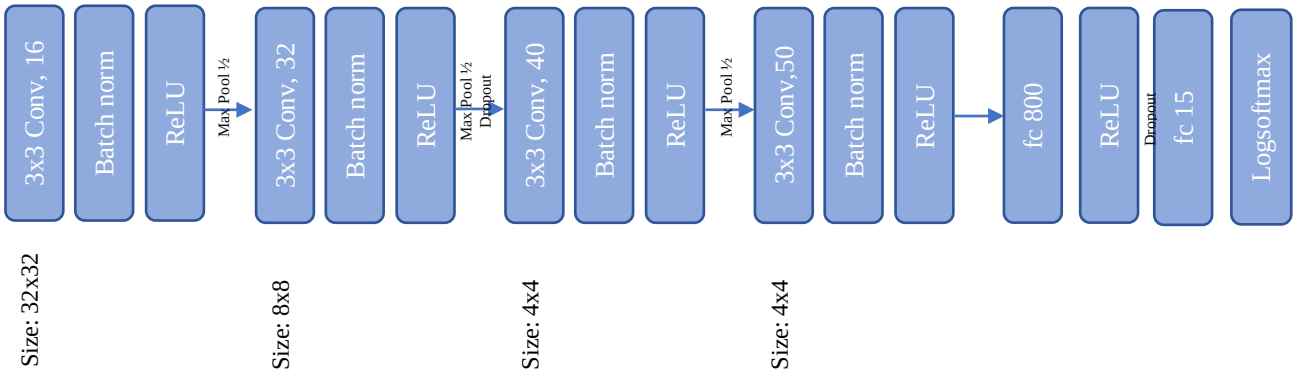
Machine learning – HW2, by: saleem kheralden 212699581

Question 1:

**Model architecture:**

The CNN consists of 4 Convolutional layers and 2 Fully connected layers

And consists of 47,917 weights




**Training procedure:**

**Data augmentation:** the train dataset consists of 100,000 training points, 50,000 of which are as is and the second 50,000 a random horizontal flip with p=.3, random rotation with 10 deg. was applied to it.

**Hyperparameters:** in the first ~300 epochs the hyperparameters were:

- Learning rate: 0.001
- Batch size: 1000
- Weight decay: 1e-5

After that I lowered the batch size to 500 which seemed to help the network with the training.

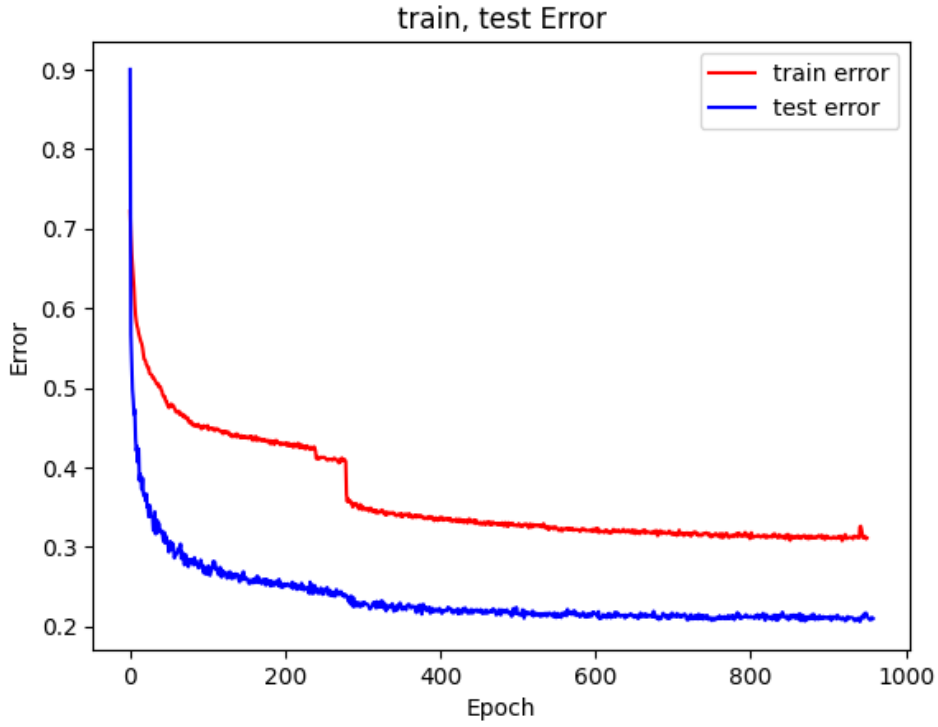
**Optimization details:** I've worked with the Adam optimizer and the learning rate stated above after, I tried the optimizer with and without the weight decay and it seemed to work better without the weight decay so eventually I set the optimizer without the weight decay.

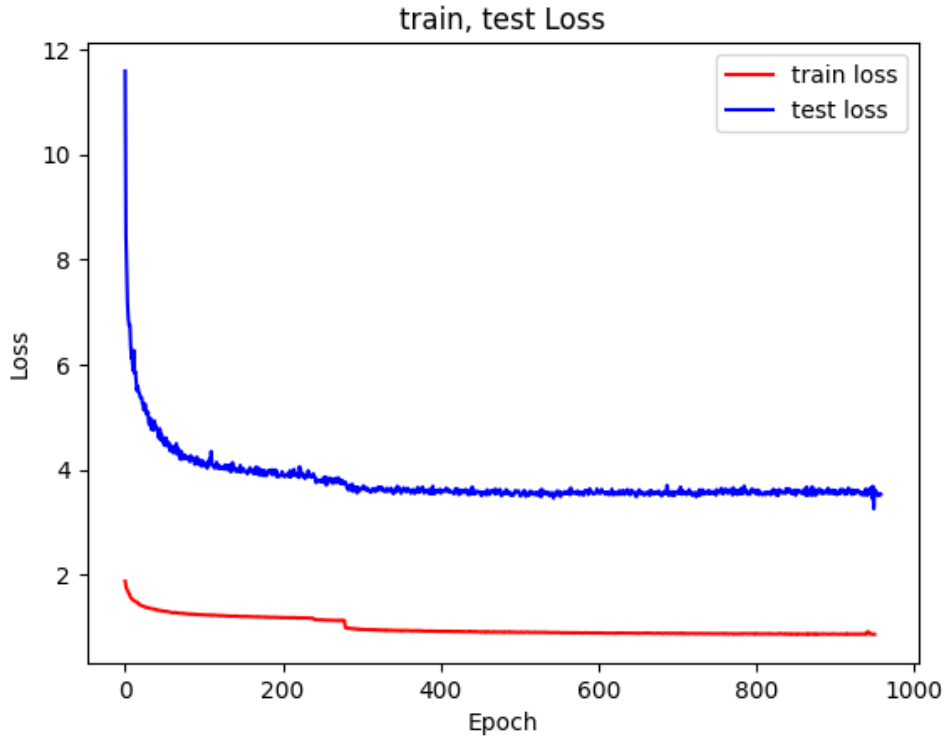
**Summary:** I worked with ~5 different architectures and almost all of them reached test accuracy 0.6-0.7 and stopped improving, while almost every time the train accuracy was lower than the test accuracy which seemed odd, almost throughout the whole training phase the train accuracy was lower the that of the test, while the loss of the train was always less, I tried to investigate what the cause was without reaching to conclusion.

**Best test error:** the best test accuracy reached (the submitted model) was 79.01% with test error of 20.99%, test loss ~3.5.

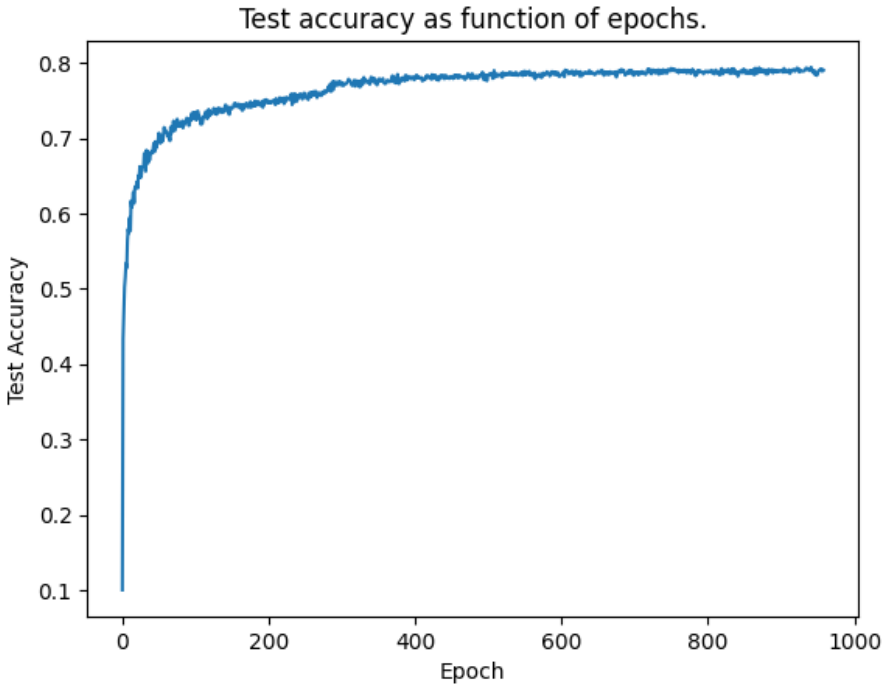
Machine learning – HW2, by: saleem kheralden 212699581

**Graphs:**



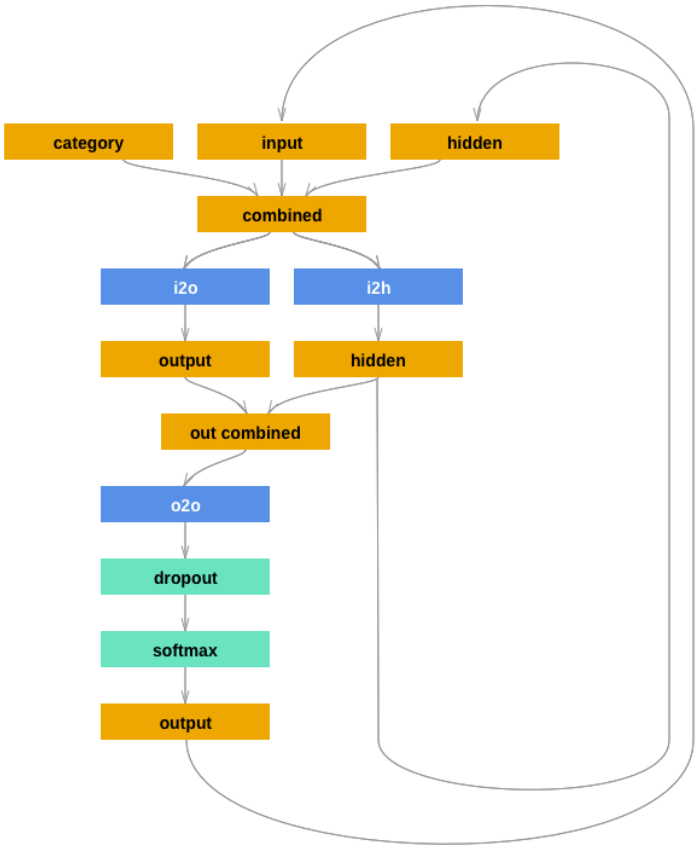
Machine learning – HW2, by: saleem kheralden 212699581



Machine learning – HW2, by: saleem kheralden 212699581

**Question 2:**

- **Model architecture**

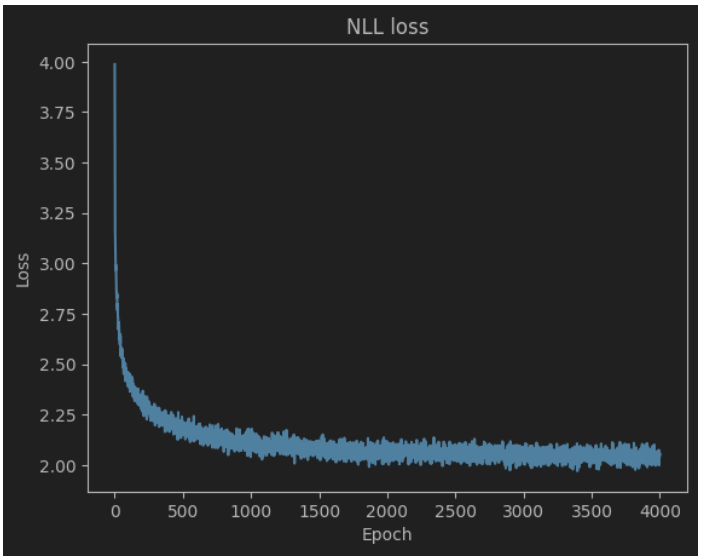



- **Training procedure:**

**Hyper parameters:** I've worked with the learning rate of .0005.

**Optimization details:** I've worked with the Adam optimizer and the learning rate stated above.

**Summary:** at first the training procedure was without an optimizer and still the network's loss decreased over time.



- **How should the input data be represented?**

The input should be the category and the first letter of the name, each converted into one hot vector and concatenated into one long vector that have two cells with 1 in it representing the category first letter pair:

Example: Category: “Arabic” would be [1, 0, …, 0, 0]

And the first letter is “a” would be [1, 0, …, 0, 0]

So the input would look like [1, 0, …, 0, 0, 1, 0, …, 0]

- **What and when do you predict?**

At each timestep given a letter and a category that’s represented as above we would predict the next letter and so on until we reach <EOS> token that’s represented as [0, …, 0, 1]

- **Samples that’s been generated by the model**

As you can see in the image below, a call to the function samples, which translates the output returned by the RNN model (tensors) into the corresponding letters, given the category and series of letters the model would return a name of each letter in the string.

As shown below after the call samples(‘Russian’, ‘RUS’) the function printed three names starting with ‘R’, ‘U’ and ‘S’.

```
samples('Russian', 'RUS')

samples('German', 'GER')

samples('Spanish', 'SPA')

samples('Chinese', 'CHI')
```

```
Romankov
Unganav
Sthovan
Grester
Estere
Roher
Sala
Panes
Alara
Awa
Bai
Chan
```