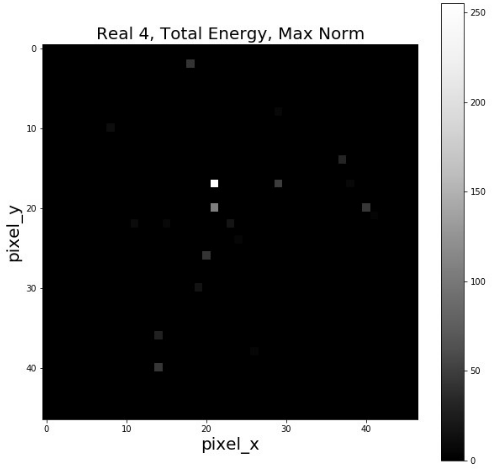
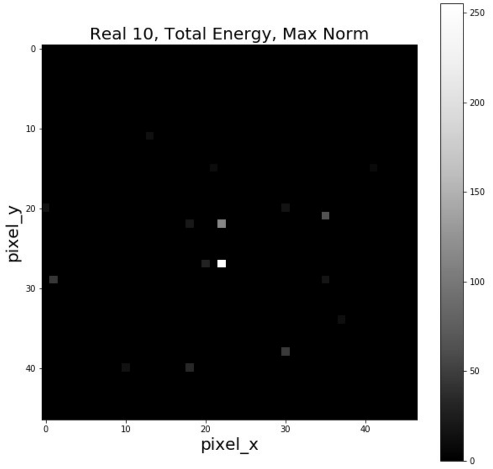
CNNs FOR ELECTRON IDENTIFICATION

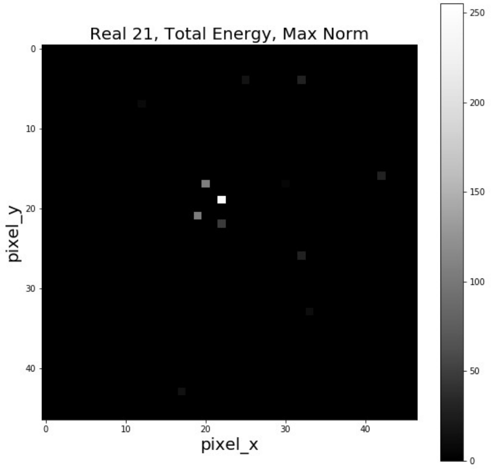
May 4th , 2020

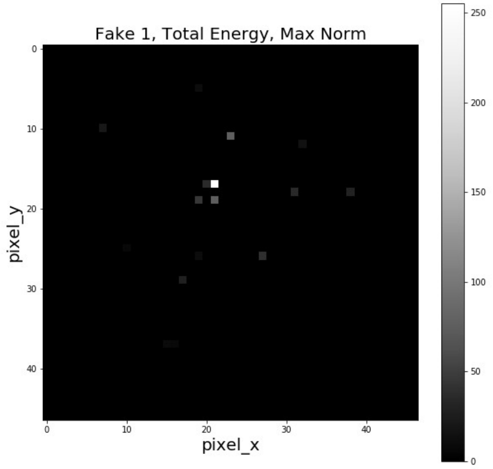
Viraj Bagal
Angira Rastogi, Sourabh Dube, Arun Thallapilil

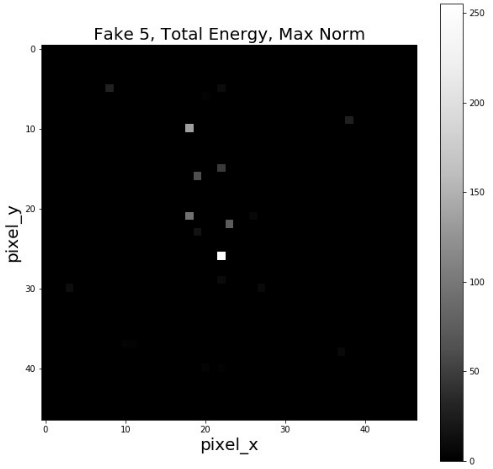
Real Vs Fake Electron Images, Max Norm Used

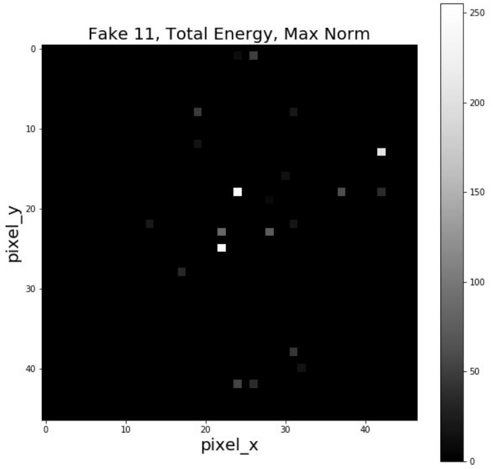




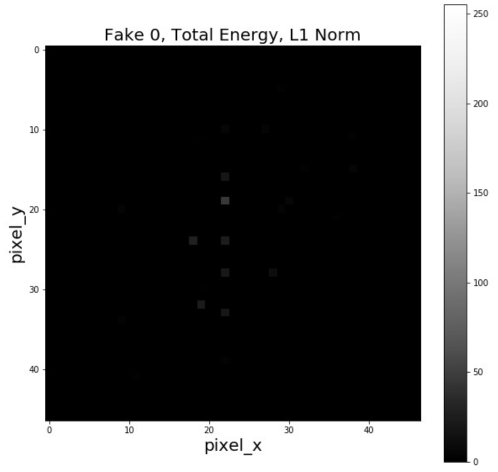


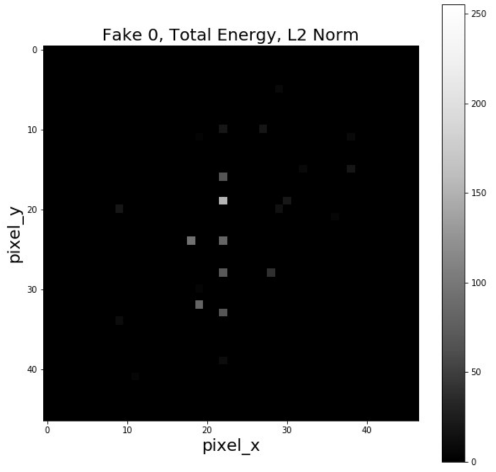


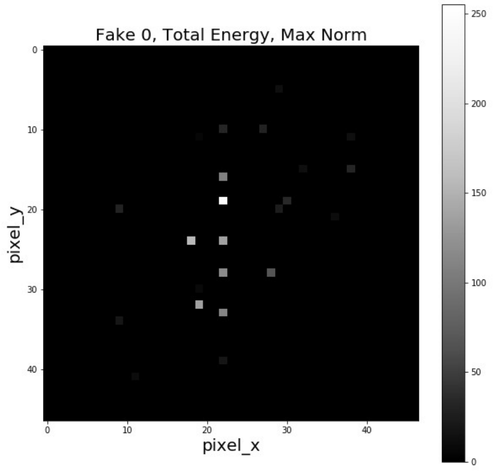


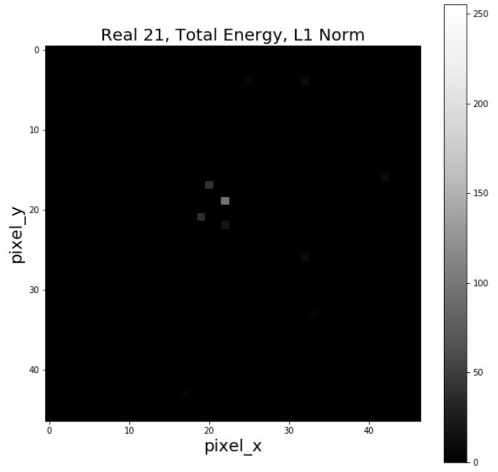


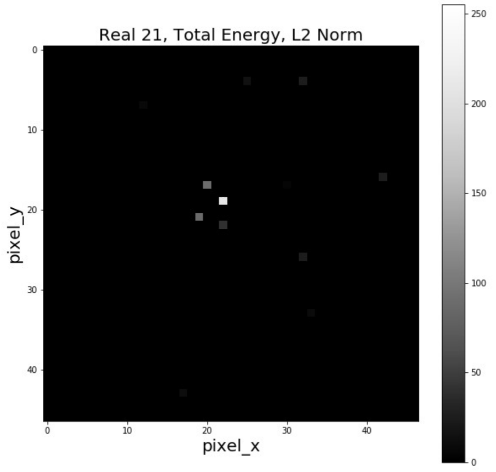
Images With Different Norms

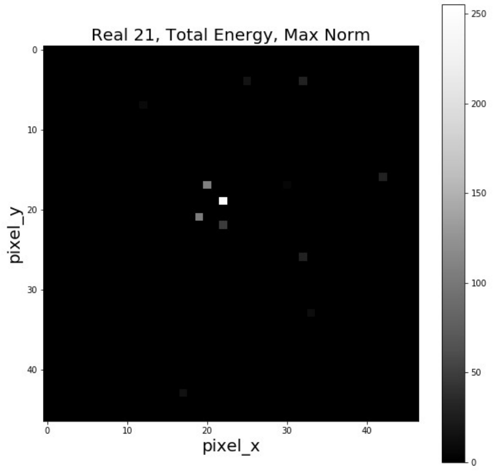










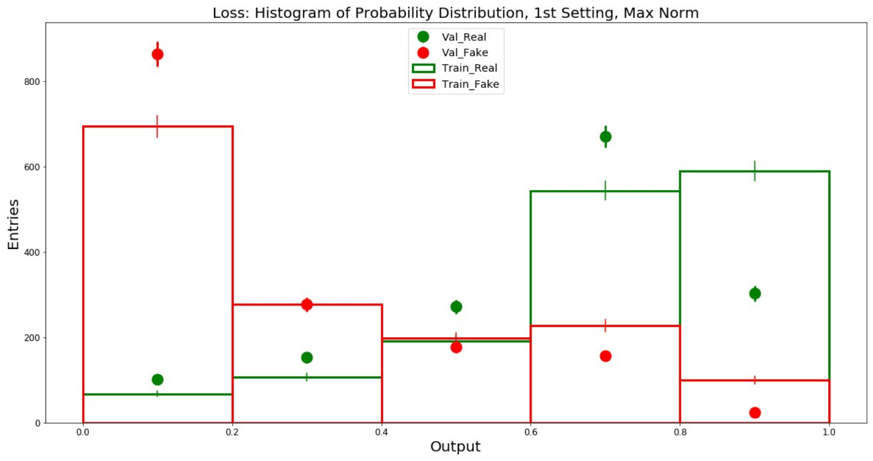


Making Of Images

1) Make 47x47 array of all 0s. 47, because we are saving calotowers within dR=0.4 of each electron and we know that $\Delta\eta \times \Delta\phi = 0.0174 \times 0.0174$.
2) Calculate $(22+d\eta/0.0174, 22+d\phi/0.0174)$ for each calotower w.r.t (η,ϕ) of the electron and make it integer. This becomes the index of the position of calotower in the array. 22 is used so that the reference electron is at the center of the array.
3) Use total energy of calotower for filling the above obtained indices of the array. Save the array

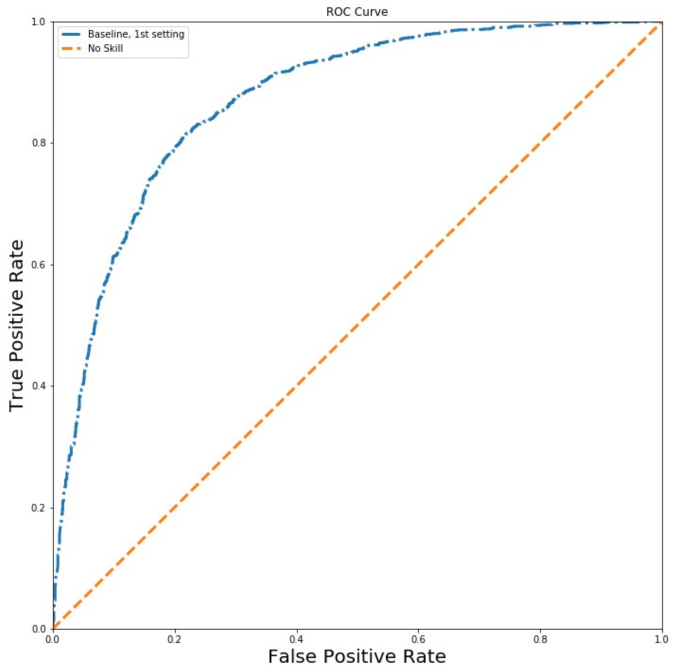
Training Pipeline

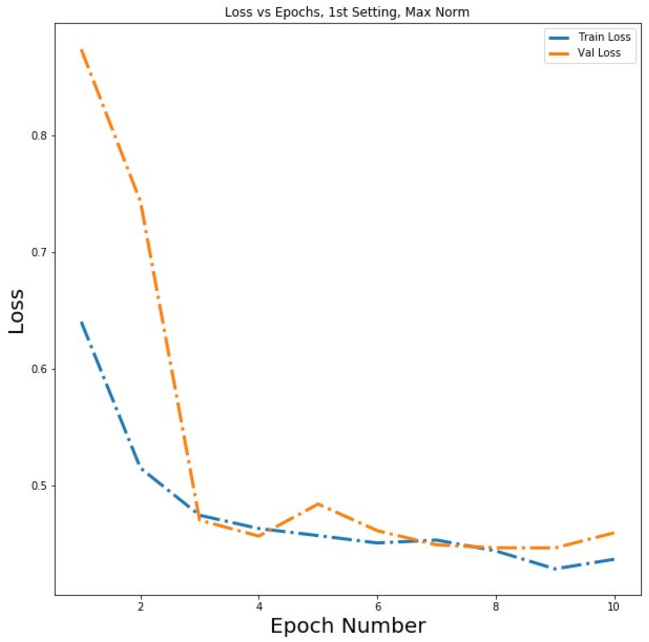
1) Instead of loading all images at once in the memory (which would lead to laptop crash), we need to generate images ‘on the fly’. So, I wrote a Dataset class for it and wrapped it inside DataLoader. DataLoader is an iterator which would generate images batchwise.
2) All the preprocessing and augmentation happens inside Dataset class. Preprocessing includes using one of the norms on the images. For now I am using only Horizontal Flip for augmentation otherwise the network is overfitting immediately after 4 epochs.
3) Baseline architecture consists of [Conv->BatchNorm->ReLU->MaxPool]x3 followed by Dense layers of 128 units. Each Conv layer has 64 filters.
4) Initially, I used Adam optimizer. Different optimizers update the weights of NN in different ways. Standard algorithm that everyone knows is the basic Gradient Descent. Unlike SGD, Adam stores the gradients of previous time steps as well to decide the direction of weight updates. It is the most widely used optimizer.



Some Evaluation Curves

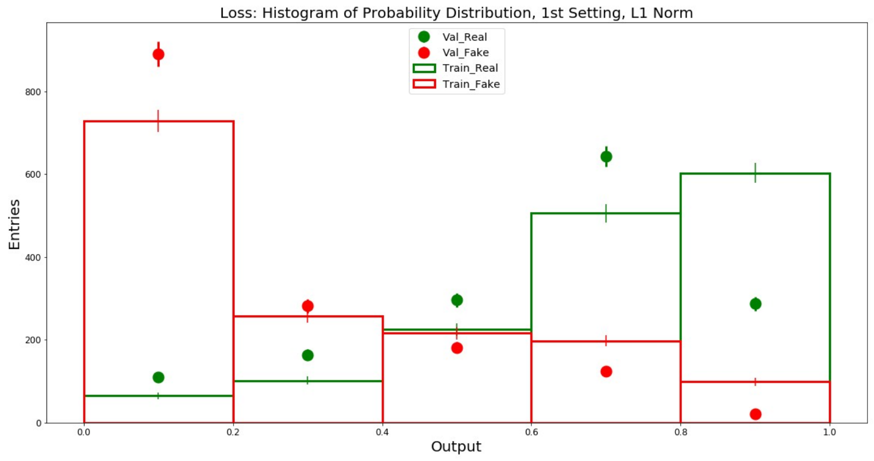
- Histogram of Probability Distribution from CNN
- ROC Curve
- Loss vs Epochs
- Binary Cross Entropy used as Loss function

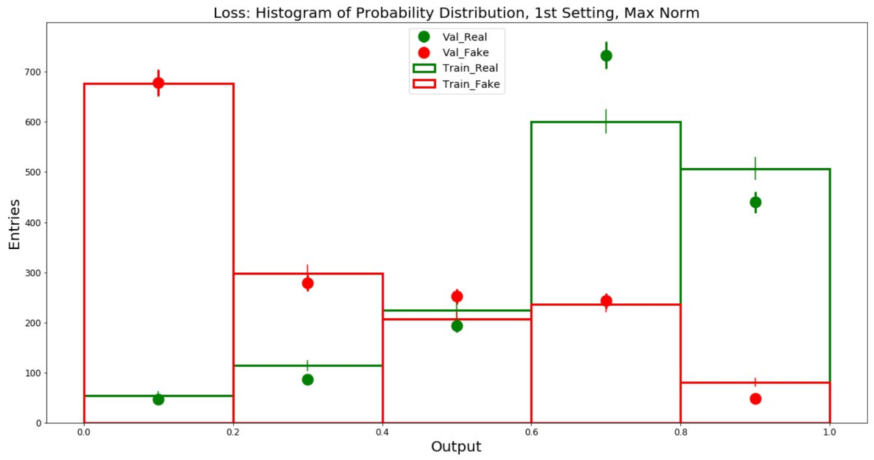


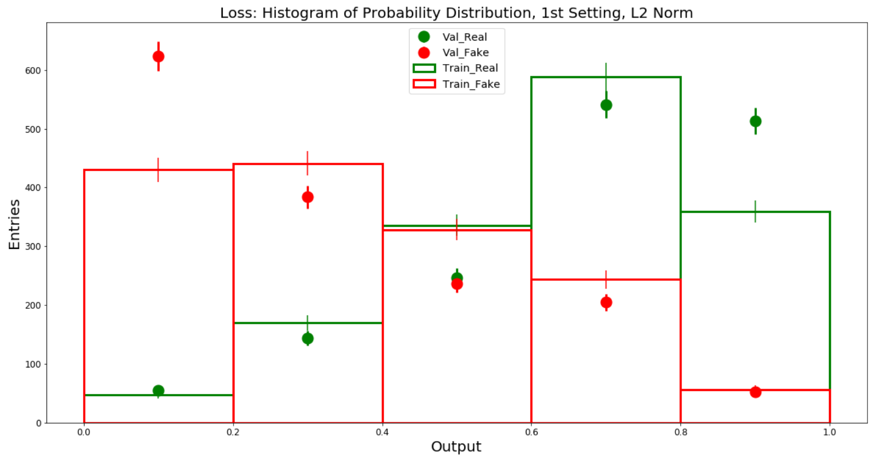


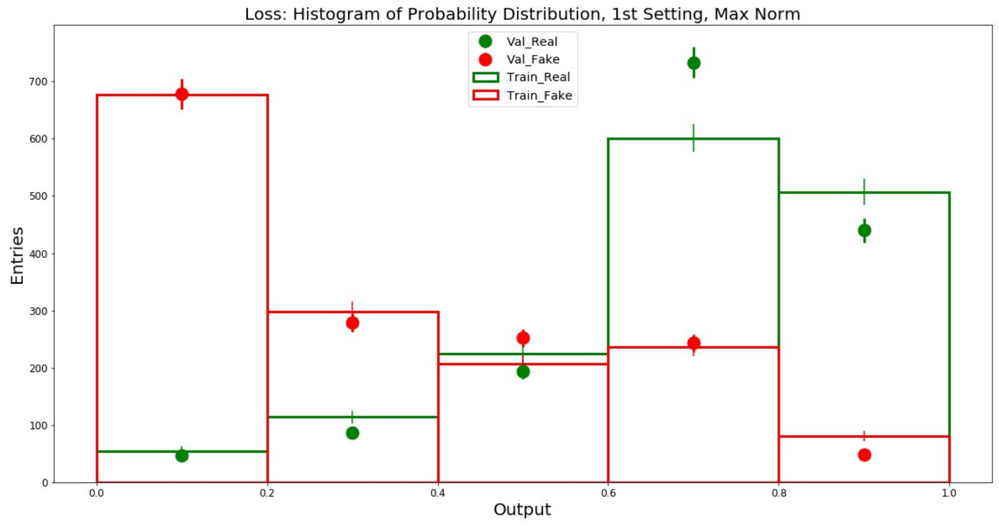
Probability Distribution Histograms For Different Norms

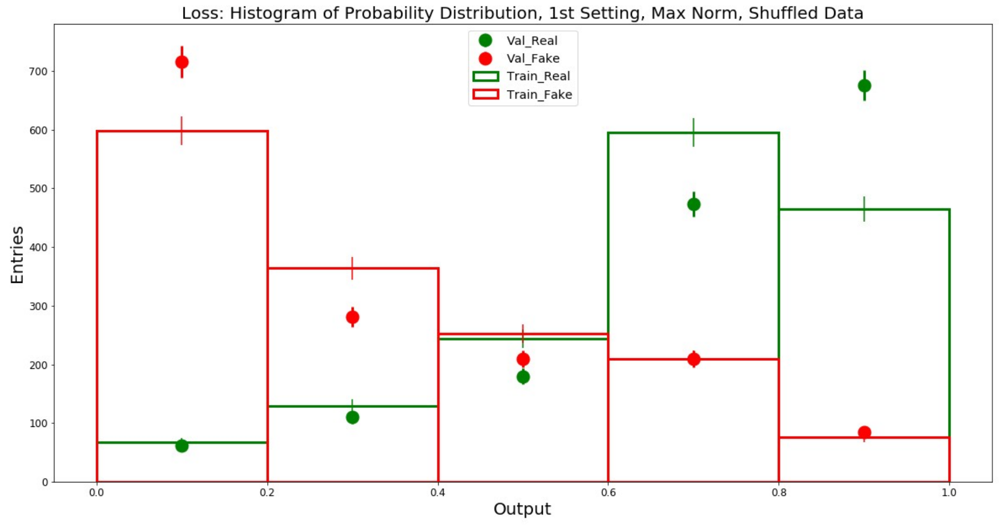
- Baseline Pipeline
- L1, L2 or Max norm applied on every image
- All other parameters in the whole pipeline are kept same





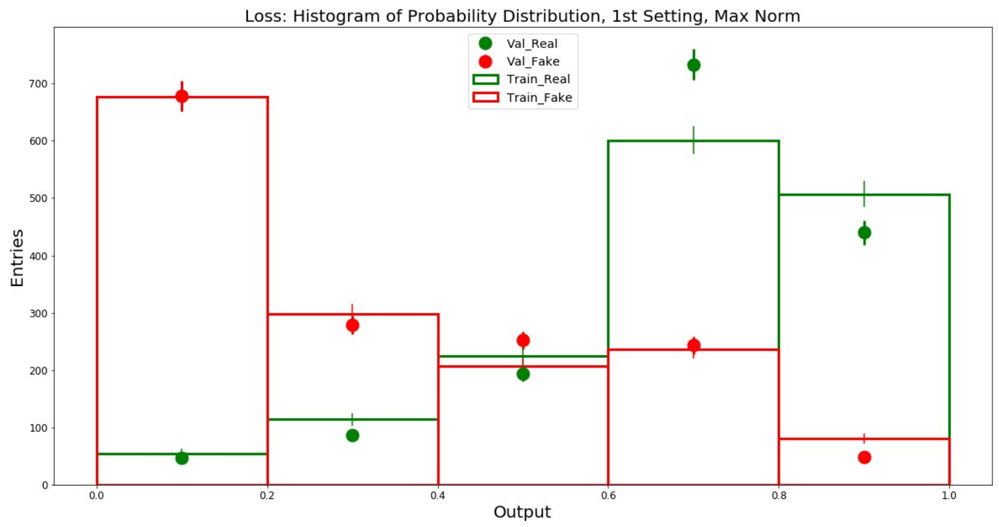






Probability Distribution Histograms After Shuffling Data

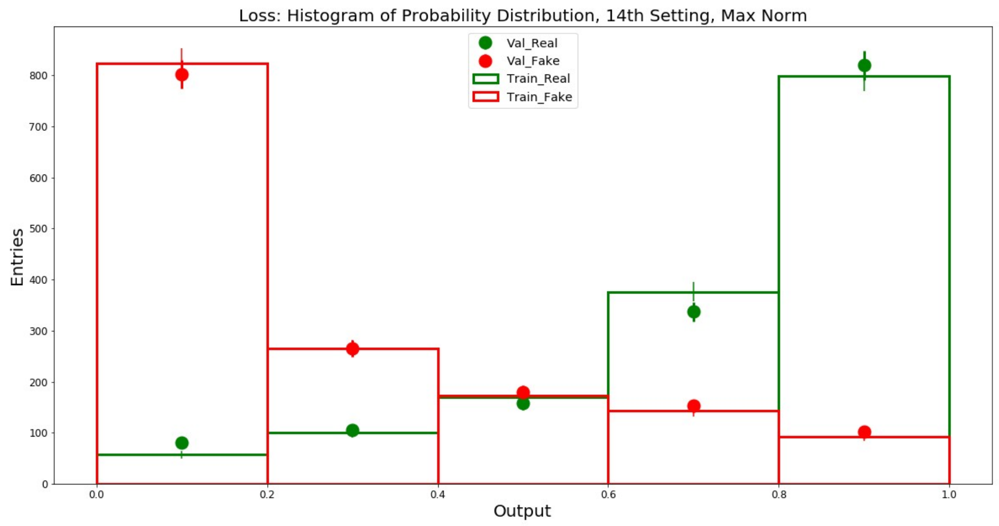
- Baseline Pipeline
- Max norm applied on every image and all other parameters are kept same
- Used **DIFFERENT** data for training and validation in both
- Used to check the consistency/robustness of model



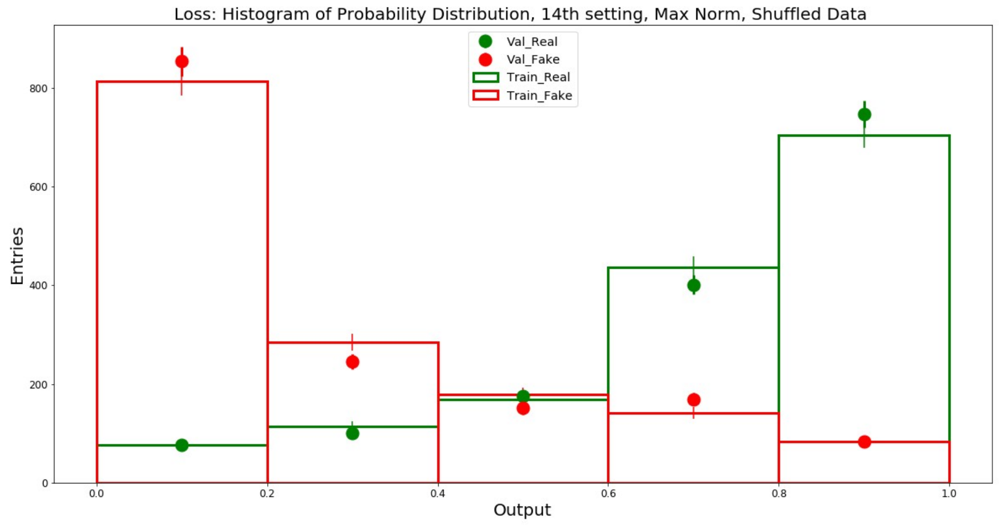
BEFORE

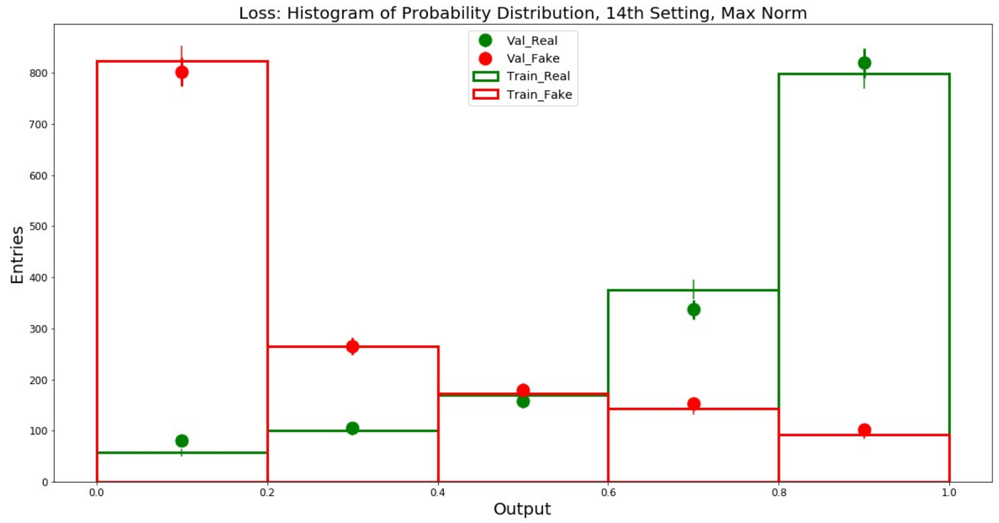
AFTER SOME CHANGES

- Change in architecture:
    - Removed BatchNorm
    - Instead of MaxPool in final layer, used MaxPool and AvgPool, and then concatenated their outputs
- Change optimizer to AdamW. AdamW is variant of Adam which applies **weight decay** on to the weights as a regularization technique



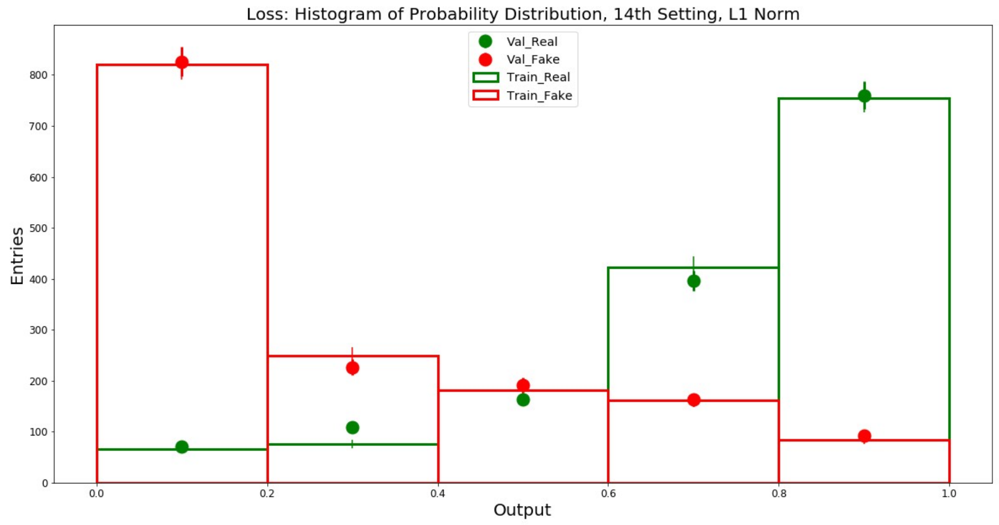
AFTER

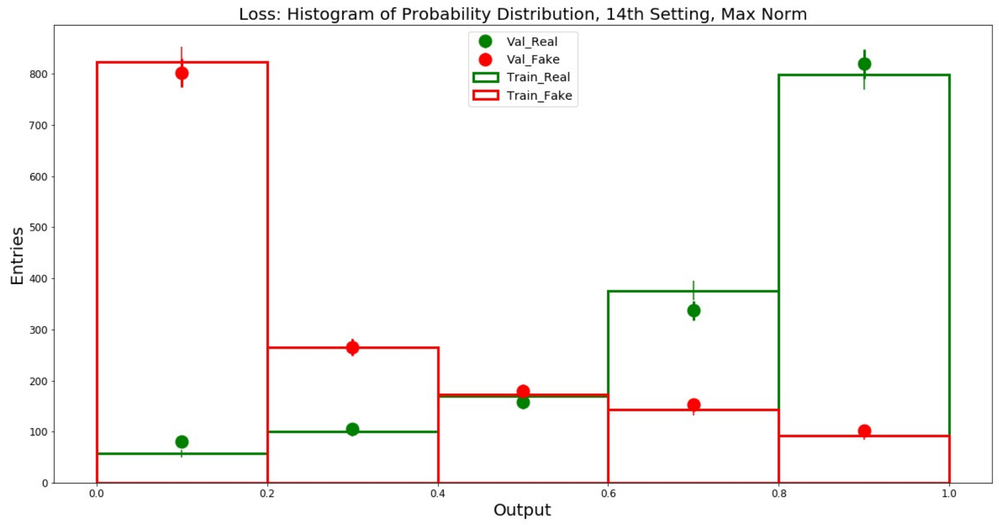




SHUFFLED DATA

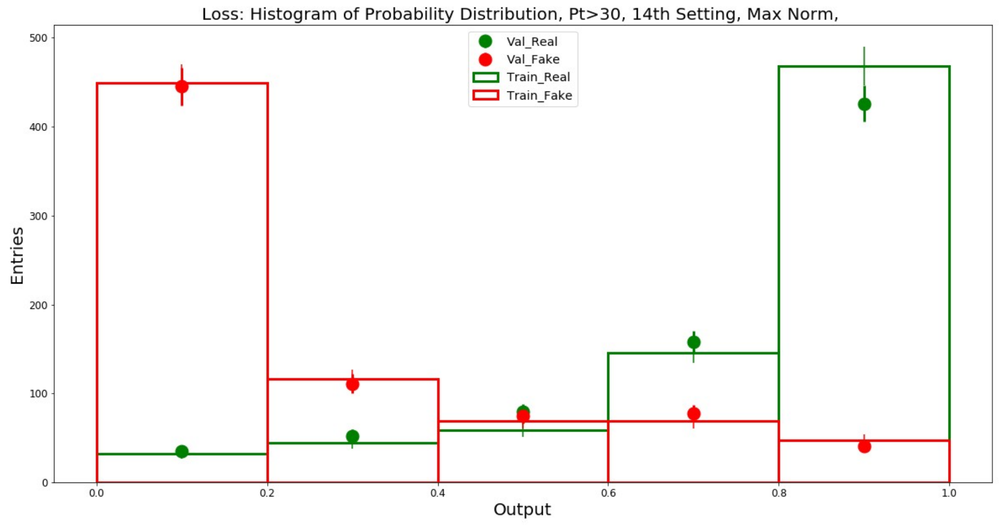
- Network Weights Fixed
- Train data, Test data changed and then just prediction done

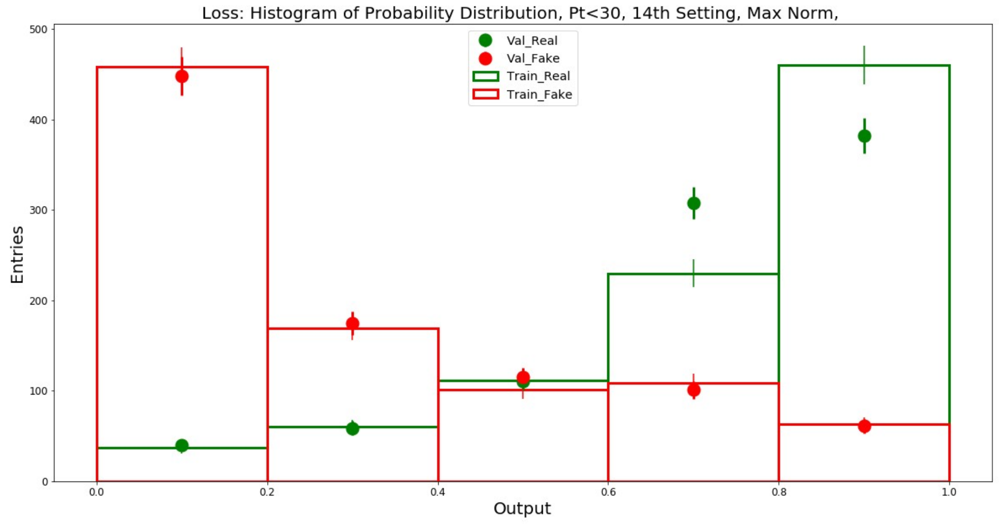




NORM CHANGED TO L1

- Network Weights Fixed
- L1 norm applied instead of Max norm and then just prediction done





Pt BINNING

- Two Bins
- Pt<30 & Pt>30
- Training done again
- Same architecture as before