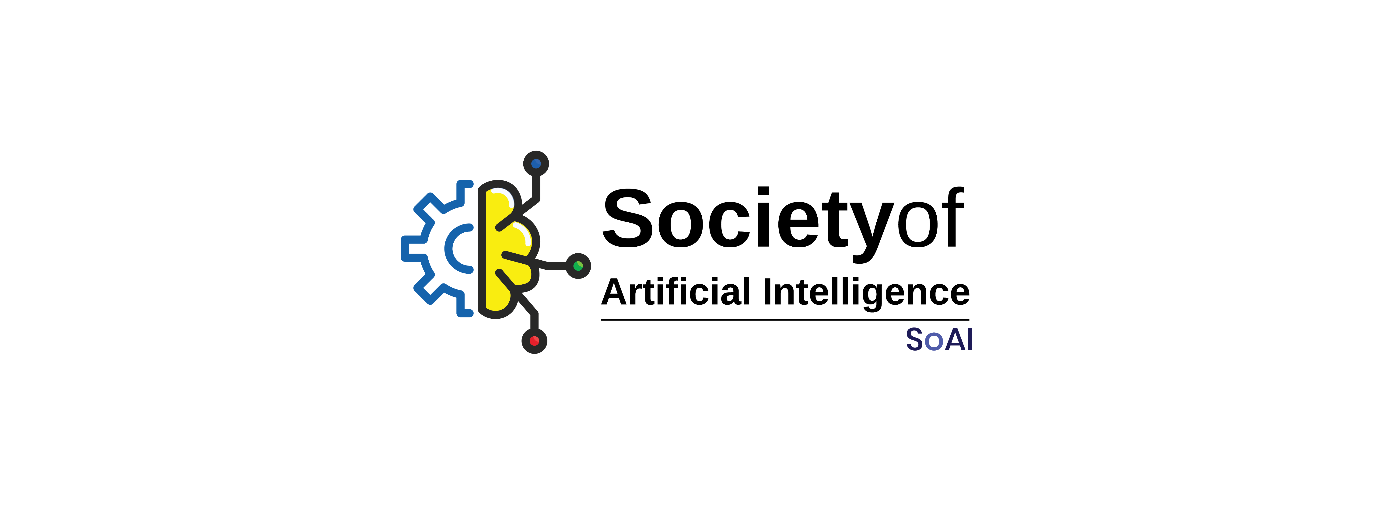
**RoadBlocks**

**BRAND NEW ARCHITECTURE FOR SIGNATURE VERIFICATION AND OVERFLOW**

****

**NEW ARCHITECTURE**

**I have read a few research papers on Using Machine Learning models to perform Signature Verification. Most of them used simple CNN and/ OR Feature Generator Approach. I had also read about how traditional methods employed the properties of the signatures to distinguish Froged from real. One of the property was the temporal aspect of a sinature. For a person who signs from left to right, the different portions of the signatured sliced vertically are correlated to each other. So if correlation of different parts of the signature in a test image do not match with correlation of different temporal parts of the real signature, then the image is forged else, it is real.**

**Building on this theory, I designed a Completely different new Architecture, in which I used one Dimensional Convolution to encode the local properties of a signature image , and the Bi-directional LSTM was used to capture the relationship of each local (vertical)slice of the image to all other slices. This way an encoded vector is produced pertaining to each image, containing the information on correlation of each local slice to other slices. Then Passing these two vectors through a Neural Network can produce a single value depicting how close are both the signatures.**

**I built and tried to train it but the network moved towards being neutral i.e. it refused to see the difference between real and forged, rather it focused on their similarities rather than their differences.**

**How did I deal with it -> Then after spending about a day training it, I hitthe idea of using Face Recognition architecture to force the network to learn the differences , instead of similarities in the signature images.**

**OVERFLOWING VALUES OF GRADIENTS**

**Firstly I designed the loss function of my final model encoder by taking MSE of the output. As a result of this the gradients came out to be huge and the ‘nan’ values began to pop up.**

**How did I come out of it -> I took sigmoid of the loss above as final loss, this prevented the overflow of the gradient values and thence, the model trained without overflow.**