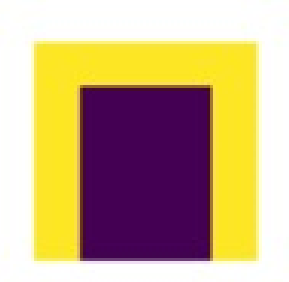
Ex2

1-2:

We implemented our network with dataset represents letters as matrix.

For example the letter ת :

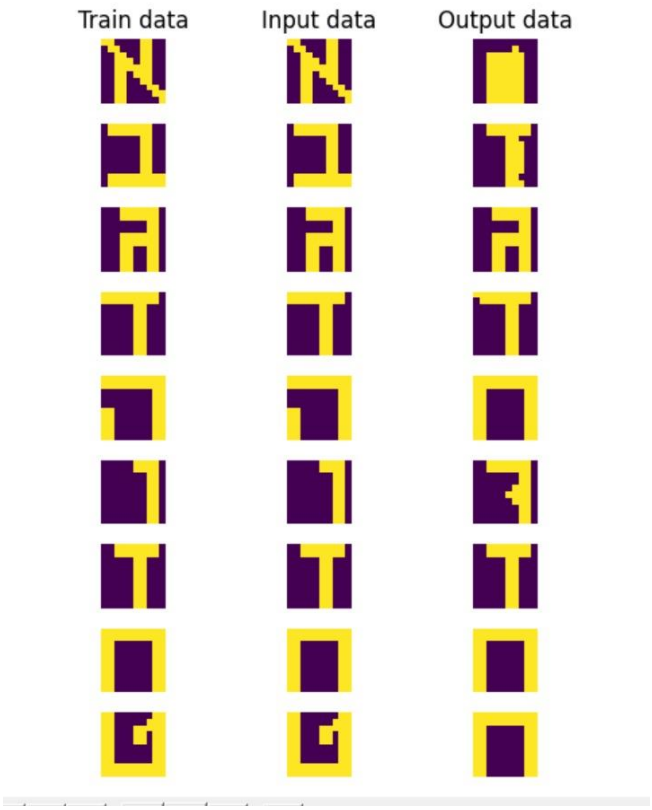


Represented as:

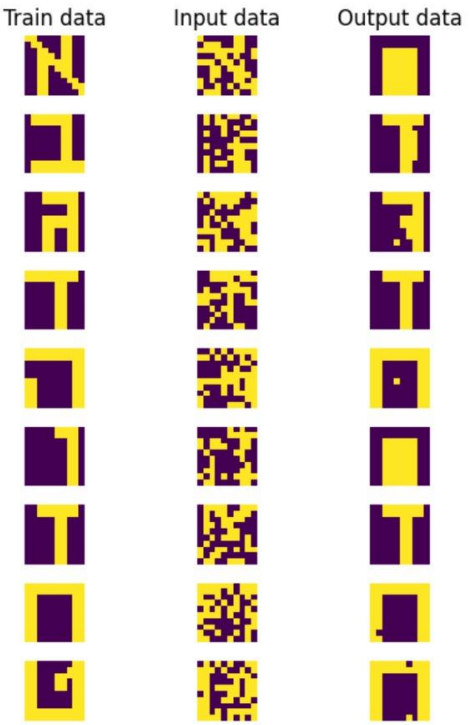
```
# ת
ת = np.array([[1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
              [1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
              [1, 1, -1, -1, -1, -1, -1, -1, 1, 1],
              [1, 1, -1, -1, -1, -1, -1, -1, 1, 1],
              [1, 1, -1, -1, -1, -1, -1, -1, 1, 1],
              [1, 1, -1, -1, -1, -1, -1, -1, 1, 1],
              [1, 1, -1, -1, -1, -1, -1, -1, 1, 1],
              [1, 1, -1, -1, -1, -1, -1, -1, 1, 1],
              [1, 1, -1, -1, -1, -1, -1, -1, 1, 1],
              [1, 1, -1, -1, -1, -1, -1, -1, 1, 1],
              ])
```

we implemented 100 neurons hopfield network model and ran it using our data.

Running example with no noise:



Example with 30% noise

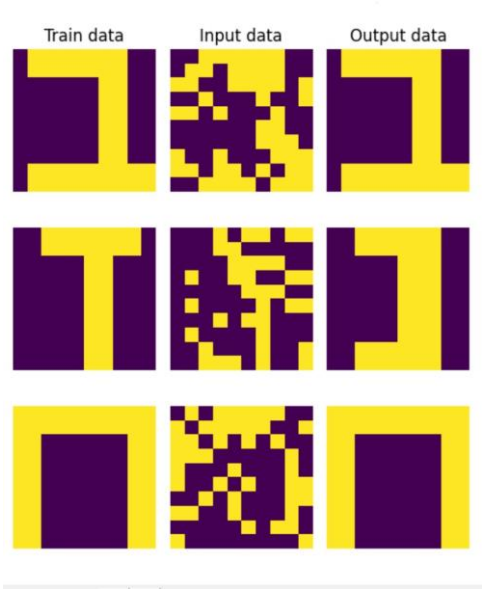


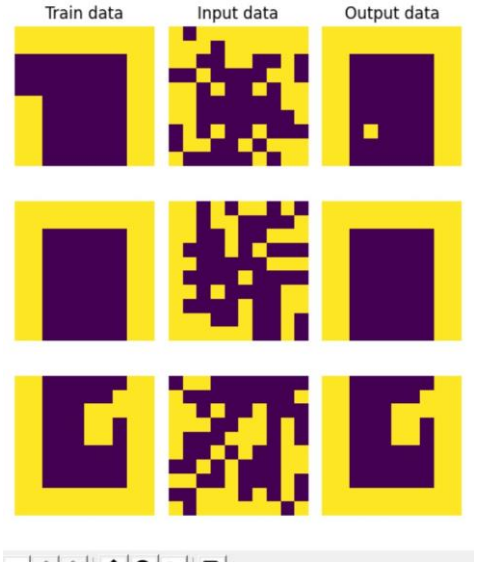
3:

As we see in the examples above not all of our pictures are stable points.

We can see that without the noise we have some stable points like ת ד.

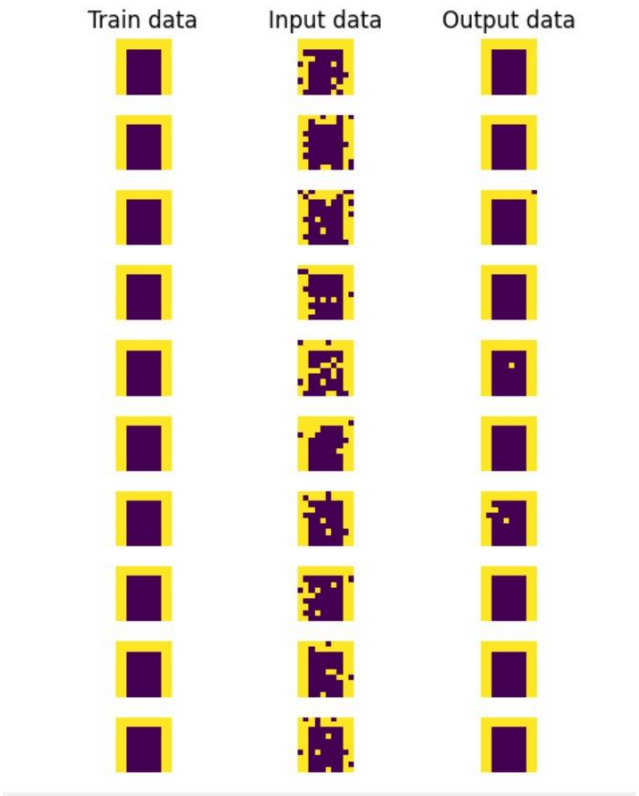
Also we can see that when we store less letters we got better results, we tried to check if there is a difference if we choose less similar letters or more similar letters but we saw that in both cases the results were better.





4:

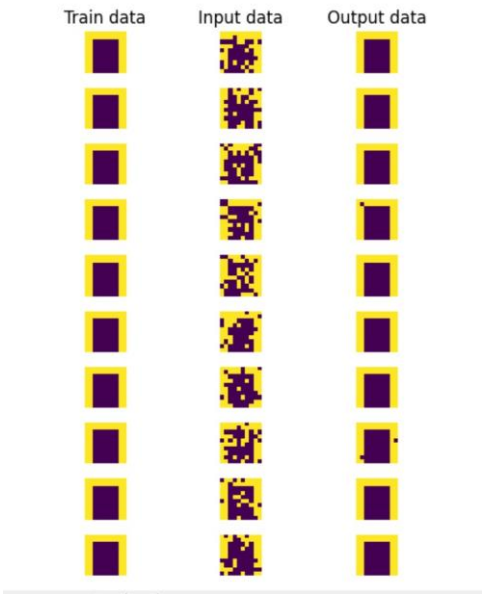
1 - We can see that in most of the runs it does converge to ɲ also we can see that even in the cases it does not converge it was very close.



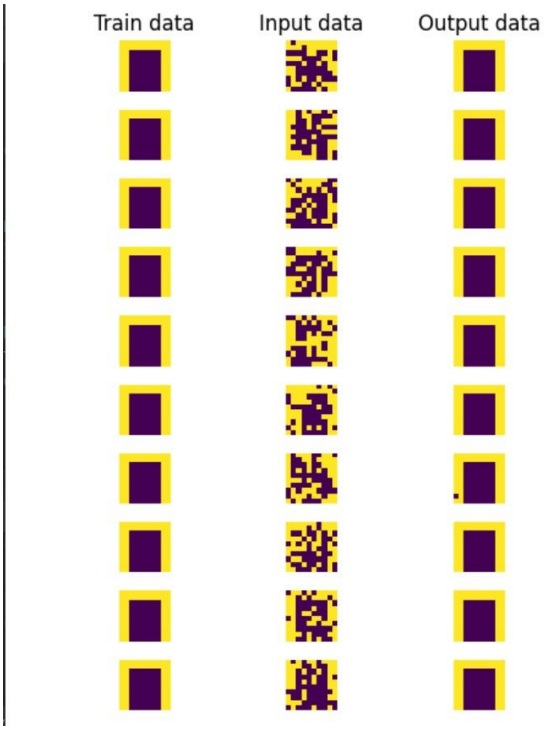
2 –

We ran it with different level of noise :

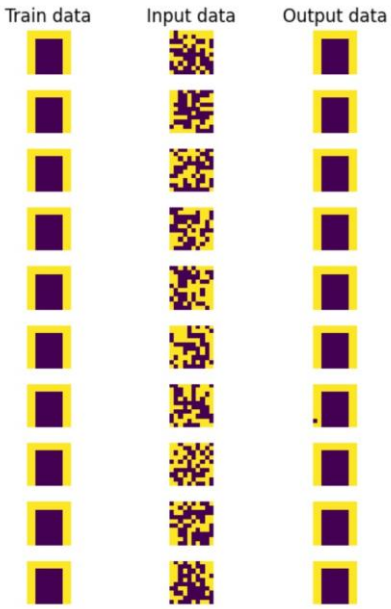
20% - almost all of them converge



30% - even better results.

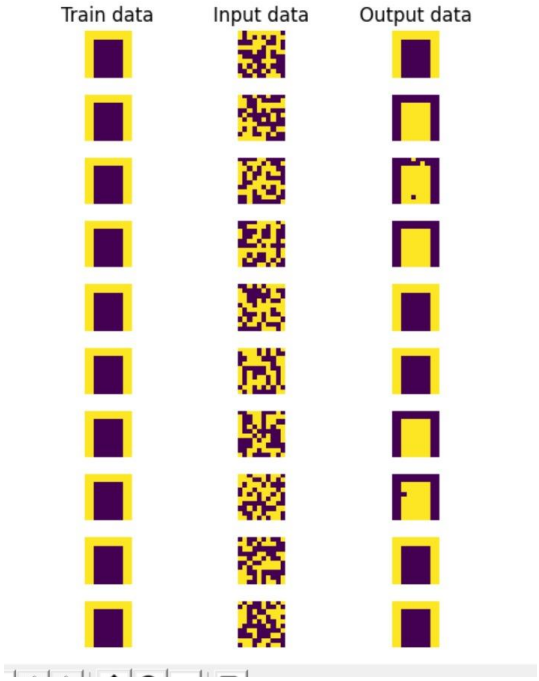


40% -

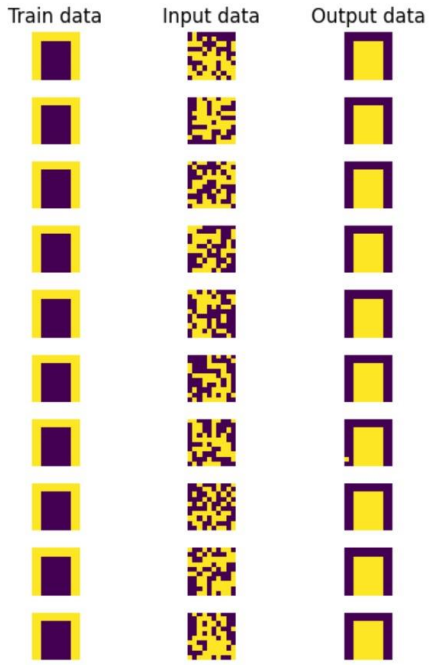


50%- now we can see that the results are getting worse.

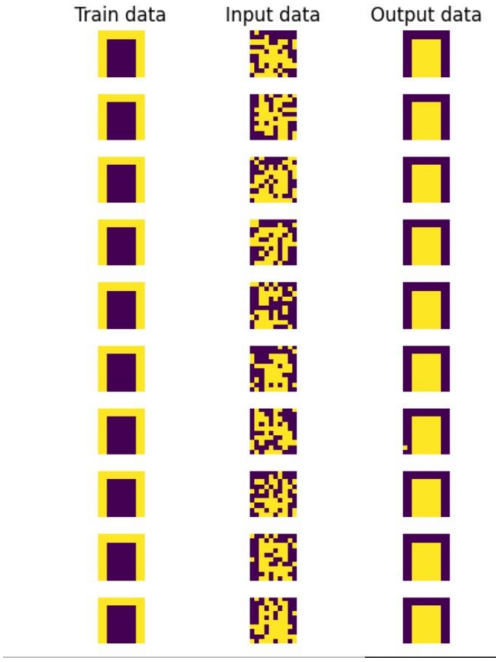
We got 50% accuracy.



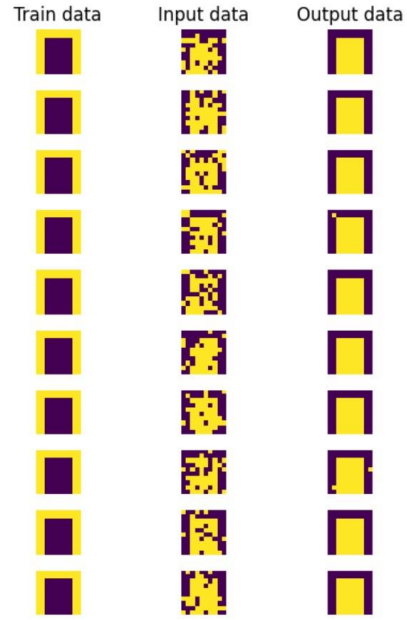
60%-



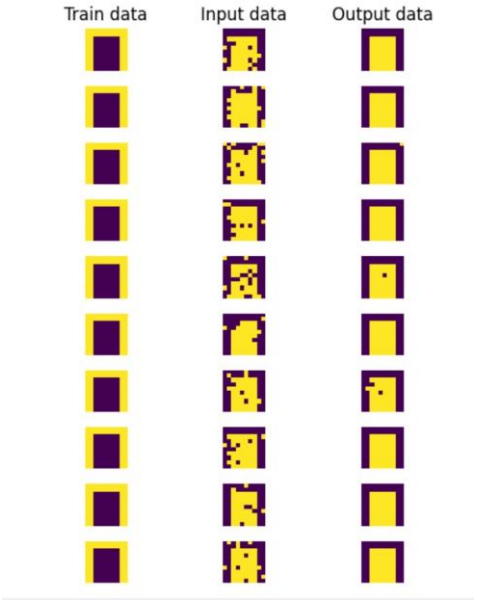
70% -


Train data
Input data
Output data

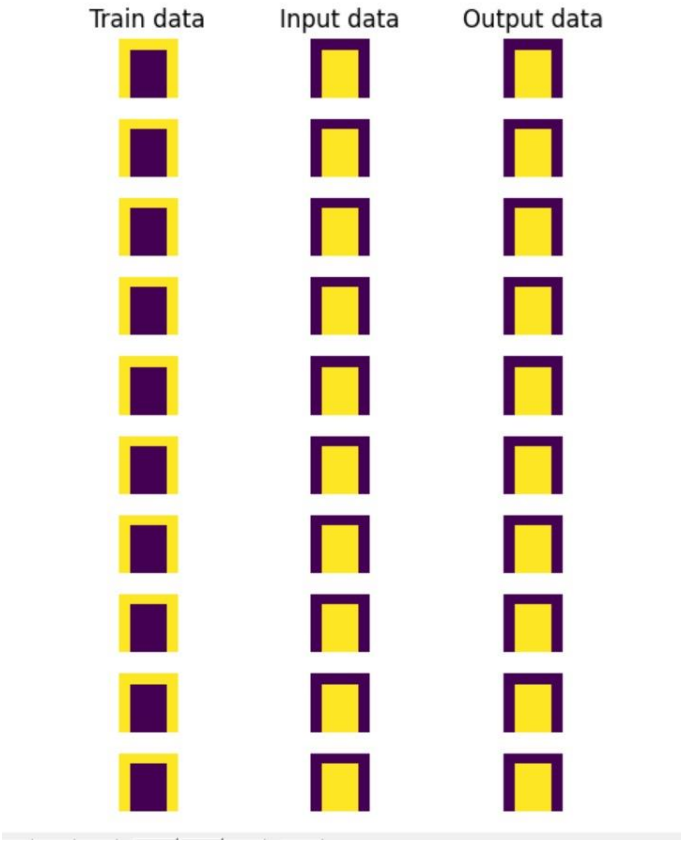
80%-


Train data
Input data
Output data

90%-


Train data
Input data
Output data

100%-


Train data
Input data
Output data

We can see that with the letter ת we get good results because the letter is very simple shaped – the letter is basically the frame of the pictures marked as 1's and all of the res as -1 so even with greater noise percentage (up to 40%)we can get relatively good accuracy because we have less transitions between values in the representing array.

we see that from 50% noise and above the results are far worse until we get ever "opposite" converge because it reverts the points and we get opposite shape of the letter 'ת' so it recognize the letter but in opposite shape.

It took 25 epochs to converge when it worked.