(5.1)

It's a RBF method with discrete kernel function. One of the disavantage of the method is that the points are less than δ will all have 0 for hidden value which will cause relatively low discriminating ability.

(5.3)

The RBF networks are conceptually very similar to K-nearest Neighbor model. The basic idea is that a predicted target value of an item is likely to be the same as other items that are close to the predicted target. So the weights of the output layer is learned in a supervised fashion.

(5.6)

This technique will have a same effect with regulization. Because it remove the effect of some activation which is noisy. Therefore, it will improve the accuracy by fitting the function appropriately on limited training set and avoid overfitting.

(7.6)

The mutations of training datasets are tagged, which means we can use a classifier to learn how to classify each position as "mutation" and "no mutation", by apply logistic sigmoid function at the output layers. And we can use the test dataset to generate a prediction.

(7.7)

① In the first type, the output sequences are directly inferred base on the phrases and clues word in the input sequences.

② The second type is using 2 RNN, one for Input layer as an encoder, the other one for output layer as a decoder. The model are with different weights.

(7.10)

For each language, create a encoder and decoder, which have the same dimensionality.

Next, we train each pair of encoder and decoder, update the weights through all training data.

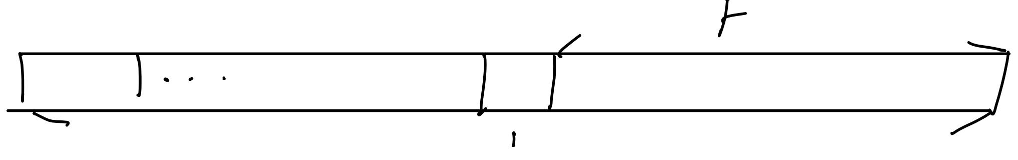
(I) For a particular sentence, all encoders can create the same universal code.

(II) We can use any pair of encoder and decoder across different languages to achieve the task of language translation.

(8.2)

① The length of output is $(L-F+1)$ #

② The padding should be $(\frac{F-1}{2})$ #



(8.7)

$$\begin{bmatrix} 4 & 6 & 4 & -3 & -3 \\ 0 & -1 & 0 & 1 & -2 \\ -5 & -6 & 1 & 1 & 0 \\ 6 & 11 & 1 & -3 & 4 \\ 3 & 3 & 4 & 4 & 2 \end{bmatrix} \#$$

(8.8)

$$\begin{bmatrix} 7 & 7 & 5 & 7 \\ 8 & 8 & 5 & 7 \\ 8 & 8 & 6 & 7 \\ 8 & 8 & 6 & 6 \end{bmatrix} \#$$

(9.2)

Neural networks have more layers than softmax regression, so it can do more complex representation learning to learn some subtle things in the data which cannot achieve by softmax regression.

(9.3)

① Why it's a more difficult problem?

Because the payoffs of the machine need to depend on the notion of states.

② Solution:

We can adopt the Bellman's equation to

pick an optimal value action for each step.

(9.4)

① Policy gradients will be more appropriate than Q-learning, because the goal of Policy Gradients is to learn a map from state to action, which can be sochastic and works in continuous action spaces.

②

Determistic policy can be adopted, because the actions value determine outcome. There is no certainty.

③

The optimal policy for human players is to choose three of them at random.