Visual Recognition

Assignment 1

Ishaan Sachdeva

IMT2018508

Task1

In this task we have added the bias variables $\mathbf{b_y}$ and $\mathbf{b_h}$ to the RNN.

Feed forward equation for RNN is given by:

$$h_t = \phi_h(W_{xh} \cdot x_t + W_{hh} \cdot h_{t-1} + b_h)$$

$$\hat{y}_t = \phi_o(W_{yh} \cdot h_t + b_y)$$

In feed forward propagation of RNN, using the input variable and previous hidden state, the current hidden state is calculated. This hidden state is used for calculating the next hidden state and output of current state as shown in the above equations. Here $\mathbf{b_y}$ and $\mathbf{b_h}$ are biases.

BackPropagation Through Time (BPTT): Gradient wrt to the weights are calculated to update the weights and minimise loss.

Gradient wrt $\mathbf{b_y}$:

$$\frac{\partial L_t}{\partial b_y} = \frac{\partial L_t}{\partial \hat{y}} * \frac{\partial \hat{y}}{\partial o_t} * \frac{\partial o_t}{\partial b_y}$$

$$\frac{\partial L_t}{\partial b_y} = (-1/\hat{y}) * \phi_o'(W_{yh} \cdot h_t + b_y) * 1$$

$$\boxed{\frac{\partial L_t}{\partial b_y} = (-1/\hat{y}) * \hat{y} * (y - \hat{y}) \quad \Rightarrow \quad \frac{\partial L_t}{\partial b_y} = -1(y - \hat{y})}$$

Gradient wrt b_h:

$$\frac{\partial L_{t+1}}{\partial b_h} = \frac{\partial L_{t+1}}{\partial h_{t+1}} * \frac{\partial h_{t+1}}{\partial b_h}$$

$$\frac{\partial L_{t+1}}{\partial h_{t+1}} = -W_{yh}^T * (y - \hat{y}) \quad \textbf{and} \quad \frac{\partial h_{t+1}}{\partial b_h} = \phi_h'(z_{t+1})(W_{hh}\frac{\partial h_t}{\partial b_h} + 1)$$

$$\boxed{\frac{\partial L_{t+1}}{\partial b_h} = -W_{yh}^T * (y - \hat{y}) * \phi_h'(z_{t+1})(W_{hh}\frac{\partial h_t}{\partial b_h} + 1)}$$

Observations after adding bias:
Loss in both the cases ie with and without biases was approximately same.

Hidden layer = 10 and input string = "RNN from scratch"

No. Of iterations	Without Bias(Loss)	With Bias(Loss)
1000	14.952	18.951
3000	1.8317	2.317
5000	0.7913	0.920
7000	0.4868	0.548
9000	0.3482	0.385

Task 2

In this part, the **SGD** gradient descent algorithm is replaced by **Adagrad** gradient descent optimisation algorithm.

The difference in Adagrad and SGD is that adagrad modifies the learning rate at each time stamp based on previous gradients.

Observations after using both the algorithms to update the model:

The Adagrad algorithm took very less number of iterations to train the model and predict the output compared to SGD. Loss in case of SGD was 0.38 whereas in case of Adagrad it 0.00002 after 9000 iterations.

No of iterations	SGD(Loss)	Adagrad(Loss)	SGD(Output)	Adagrad
1000	18.951	0.0948	RNNcr a scmtofrfr	RNN from scratch
5000	0.920	4.26191E-05	RNscrct tch N fro	RNN from scratch
9000	0.385	1.9888E-05	RNN from scratch	RNN from scratch

HIDDEN LAYER = 10

Task 3

In this task, the model was trained with various vector sizes of hidden layer.

Observations after increasing the size of hidden layer are:

1) It took less number of iterations to predict the output

2) loss decreases as size increases.

vector size of hidden layer	Loss(after 9000 iterations)
10	0.38587279
15	0.23101679
20	0.15662216