CSE628 – Assignment 3

Gaurav Ahuja
SBU Id: 111500318

1. **Experiments and results**
   In all the experiments, training of the model was done for 1000 steps.
   For the final submission, the model was trained for 2000 steps.
   - I experimented with power 5. I did not experiment with power 4 as power 4 polynomial is a symmetric function with respect to x axis. Power 5 is similar to power 3, just rises more sharply. It actually produced better results than power 3 polynomial.
   - Using two and three hidden layers do not lead to increase in accuracy. In fact, using 3 layers (100 units in last 2 layers) leads to worse accuracy as compared to the single hidden layer network.
   - In terms of non-linearities, sigmoid performs the worst. ReLu, tanh and power 3 perform nearly the same. In my experiment, the order in terms of accuracy is:
     Sigmoid < ReLu < Power 3 < tanh < Power 5
     This order is on one set of experiment. It needs to be run multiple times to concretely establish the order of these non-linearities.
   - In experiment 2b,
     hidden = f(w1*embed_words) + f(w2*embed_pos) + f(w3*embed_label)
     out = softmax(w4*hidden)
     The network was not allowed to look at words and pos tags in the initial layer. It looked at the summation of these representation to produce the output. The accuracy was slightly lower as compared to the default configuration.
   - In experiment 2c, word embeddings were not trained during backpropogation. The pos and label embeddings were initialized with one-hot vectors. A new function load_embeddings_one_hot was implemented for this task. The accuracy was lower as compared to the default configuration. As stated in the research paper, label embeddings do not affect the results much, but pos embeddings help in achieving better results.
   - In experiment 2e, gradients were not clipped. Gradient clipping scales down the gradient so that the norm of the gradient is less than or equal to the clipped value. Gradient clipping prevents the problem of exploding gradients. If gradients are large, taking a step in the direction of gradients can lead the loss function to diverge instead of converging. Hence gradient clipping is helpful.
     If gradients are not clipped, the loss becomes NaN in less than 100 iterations in the experiment.

Experiment	Config	UAS	UASnoPunc	LAS	LASnoPunc	UEM	UEMnoPunc	ROOT
Default	Default	65.967	69.27	61.0589	63.835	7.941	8.47	48.882
1	2 Hidden layers	64.538	68.258	59.371	62.81	6.47	6.882	42.294
1	3 Hidden layers	50.532	54.14	39.195	41.708	2.117	2.235	12.176
2a	Sigmoid	52.648	55.708	43.744	46.035	3.529	3.647	35.705
2a	Tanh	66.809	70.007	61.996	64.779	7.647	8.176	51.47
2a	ReLu	65.386	68.493	60.6	63.414	7.058	7.411	49.47
2a	Power 5	67.053	70.347	62.32	65.282	7.352	8	47.352
2b	No skip Connection	65.204	68.493	61.021	63.96	7.176	7.647	43.294
2c	Fixed embed, One Hot pos And label	61.397	64.912	56.155	59.435	4.647	4.941	32.764
2e	No gradient Clipping	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Illustration 1: Results of experiments

2. Lessons learned

I learned some very important lessons by doing this assignment.

- **Regularization:**

  Initially, I was only adding L2 norm of weights to the loss function. My loss function was not changing much and the accuracy was also not changing. When I increased the standard deviation of randomly initialized weights, the loss increased and then decreased to a low value and nearly 60% accuracy was achieved. This happened for a very small range of variance (0.2 to 0.25). If I increased the variance, the loss kept increasing.

  Then I came to know that we had to include L2 norm of embeddings in the loss. After making this change, the network trained properly.

- **Early stopping can be good:**

  In some of the experiments, the validation accuracy at 800^{th} step was significantly more than that at 1000^{th} step, even though the loss kept decreasing till 1000^{th} step. For the best model, the validation error was decreasing till 2000^{th} step. Hence the model was allowed to train for 2000 steps.

- **Deeper network does not necessarily lead to better accuracy:**

  I used to think that a deeper network is always better than a shallower network. While it is true in theory, in practice, it is difficult to train deeper networks. It happened in this assignment that 3 hidden layers are actually worse than 2 and 1 hidden layer.