Deep Learning DS-GA 1008 Assignment3

Xiao Jing xj655

March 2019

1 Dropout

https://www.overleaf.com/project/5c86a45cbd4dbc5c7d08d4c0

1. Dropout module in torch.nn

m = torch.nn.Dropout2d(p=0.2) Here, the skipping unit possibility is 0.2 and all

2. What is it and Why is it useful? The key idea is to randomly drop units (along with their connections) from the neural during the training. network during training.

This prevents units from co-adapting too much. During training, dropout samples from an exponential number of different "thinned" networks. At test time, we just need to multiply the w with probability P.

2 Batch Norm

1. What is Mini-batch refer to deep learning?

Mini batch is one of the data sets and is a portion divided from the very large original data set.

2. What is Batch Norm and why it's useful?

Batch normalization is to limit co-variate shift by normalizing the activation of each layer (transforming the inputs to be mean 0 and unit variance). This, supposedly, allows each layer to learn on a more stable distribution of inputs, and would thus accelerate the training of the network. Also, to make the training model working well on different distribution data.

Why it's useful ? If an algorithm learned some X to Y mapping, and if the distribution of X changes, then we might need to retrain the learning algorithm by trying to align the distribution of X with the distribution of Y. For example, we train our data on only black cats' images. So, if we now try to apply this network to data with colored cats, it is obvious; we're not going to do well. The training set and the prediction set are both cats' images but they differ a little bit. In other words.

3 Language Modeling

(a)

(b)

During each train function to train each batch size sequence, the loss.backward() in line 159 is to BPTT process, using the critieria of nn.CrossEntropyLoss to find the gradient descent. Iterate training the sequence and each time use backprop to increase the model.

(c)

Wrap hidden states in new Variables, to detach them from their history. In each evaluate and each training, the hidden will be repackaged.

(c)

1. One-hot encoded vectors are high-dimensional and sparse. In a big dataset this approach is not computationally efficient.
2. High similarities between words will be found in a multi-dimensional space. Embedding allows us to visualize relationships between words, but also between everything that can be turned into a vector through an embedding layer.

(d) This work is a collaberate work, I and my team member train a model and share the model result

LSTM Epoch 20: time 46.05s — test ppl 116.47
GRU Epoch 20: time: 27.60s — test ppl 150.20
LSTM Epoch 40: time 46.05s — test ppl 109.57
GRU Epoch 40: time 27.31s — test ppl 145.92
LSTM bptt 35: time 48.06s — test ppl 109.57
GPU bptt 35: time 39.19s — test ppl 134.87
LSTM bptt 50: time 41.68s — test ppl 107.40
GRU bptt 50: time 38.72s — test ppl 114.85
LSTM emsize 100: time 28.03s — test ppl 108.89
GRU emsize 100: time: 38.38s— test ppl 218.64
LSTM emsize 200: time 28.55s — test ppl 107.58
GRU emsize 200: time 39.12s — test ppl 134.87

As to epoch, GRU just finished more quickly, but more perplexity

As to bptt, GPU run faster, but more perplexity

As to emsize, GPU run slower, and also more perplexity

(e)

To give a good effective test score otherwise, each loss will be the same

4