CS5841/EE5841 Machine Learning

Lecture 12a:NN notes

Evan Lucas

Loss functions

- Cross Entropy
  - Includes softmax(), just use linear output layer
  - For multi-class classification
- Binary Cross Entropy
  - For single output neuron binary classification
- Mean Squared Error
  - Regression, for single output neuron
- Others

Wait, when do we use softmax?

- Don’t include it in your network because it’s already included in pytorch’s cross entropy loss definition
- Inference
  - If you just want top predicted class - it doesn’t matter
  - If you want top-k or prediction probabilities, compute outputs and run through

Why shouldn’t we include softmax()?

- Numerical stability
  - underflow - really small numbers get approximated as zero
  - overflow - really big numbers get approximated as infinity

What comes out of a neural network?

- Logits
  - Raw, unnormalized values
- If multi-class classification, these can be treated as independent classifier outputs
  - This is why softmax is helpful - normalizing w.r.t. other class predictions
- After normalization - prediction probability distribution
- What is the probability distribution of labels?

PyTorch Datasets and Dataloaders

- use next(iter(myDataloader)) to see the output of the dataloader for debugging purposes
- PyTorch has some good tutorials for building custom Datasets as subclass of PyTorch Dataset
  - There's also a lazy way…
-

Lazy dataset creation (won't always work)

```
import torch.utils.data as data_utils


train = data_utils.TensorDataset(features, targets)
train_loader = data_utils.DataLoader(train,
  batch_size=50, shuffle=True)
```

Easy test/train split in pytorch

```
train_set, val_set =
  torch.utils.data.random_split(dataset, [train_size,
  test_size])
```

Other ways to specify split

Questions + Comments?

Resources used

http://cs231n.stanford.edu/slides/2023/lecture_2.pdf