50.021 -AI

Alex

Week 03: Basics of neural networks

[The following notes are compiled from various sources such as textbooks, lecture materials, Web resources and are shared for academic purposes only, intended for use by students registered for a specific course. In the interest of brevity, every source is not cited. The compiler of these notes gratefully acknowledges all such sources.]

Coding Homework – Generating Star Trek Dialogues

You can take as starting point the tutorial:
`https://pytorch.org/tutorials/intermediate/char_rnn_generation_tutorial.html`,
then use
`star_trek_transcripts_all_episodes_f.csv` as input.
What do you have to modify?

- you have only one category, there will be no category index as input
- use an LSTM, replace the custom RNN (what works well: 2 layers of LSTM 100 or 200 hidden dimensions)

reading the csv:

```
category_lines=dict()
with open('./star_trek_transcripts_all_episodes_f.csv','r') as f:
  for line in f:
      v=line.strip().replace('=','').replace('/',' ').replace('+',' ').replace('(',' ') \
        .replace('[',' ').replace(')',' ').replace(']',' ').split(',')
      for w in v:
        if (w not in filterwords) and (len(w)>1):
          category_lines['st'].append(w)
```

This gives you a dictionary with only one key and that key contains a huge list with many star trek TOS words.

You can use also the raw data `star_trek_transcripts_all_episodes.csv` but there you may need to filter more special signs. You are allowed to manually clean the csv.

What else to program

- implement a character level RNN for the star trek monologues. Model suggestion:

  - LSTM 2 or 3 layers, 100 or 200 units
  - dropout with 10% probability
  - fully connected (hidden units $\rightarrow$ number of output tokens)
  - if you use NLLLoss, then a logsoftmax layer
- implement a temperatured sample
- after every epoch sample 10 to 20 different samples. capture that output (e.g. in a .txt-file)
- measure accuracy on the test set every epoch
- copy off that .txt file and the model before you shutdown the AWS session.

What to deliver?

- your code, your trained model (can save the model dictionary of parameters rather than the model as a whole).
- the .txt file of outputs of the sampling for your epochs
- plot a graph of train and test loss over epochs
- report accuracy per epoch.
- sampling: what works well for me is a temperature of 0.5. You can use a subset of all letters as starting letters. I suggest only CAPITAL letters, no Q or XY.
- **Report the best quote you have seen !** We will vote your quotes with a price in class.

```
best_model_wts = net.state_dict()
torch.save(best_model_wts,'./model.pt')
```

Going beyond

THIS IS NOT PART OF THE HOMEWORK.

You can use your saved model to play with the sampling temperature.

The character level rnn gets words right, but not really their ordering. Since star trek uses 20th century english, you can judge that. You can try on your own to do a word-level modelling with word embeddings, but that will be much harder to achieve (one needs usually larger corpora). Modern approaches combine both ideas. One can learn hierarchical rnns (tree-LSTMs for example, where folding of batches is more complicated). You can use your code to play with.

Babbling words is not so bad and can make one achieve a lot,
compare with successes of certain politicians on other continents.