Assignment 3

Jungbeom Lee

Electrical and Computer Engineering
Seoul National University

http://ailab.snu.ac.kr

Assignment Objectives

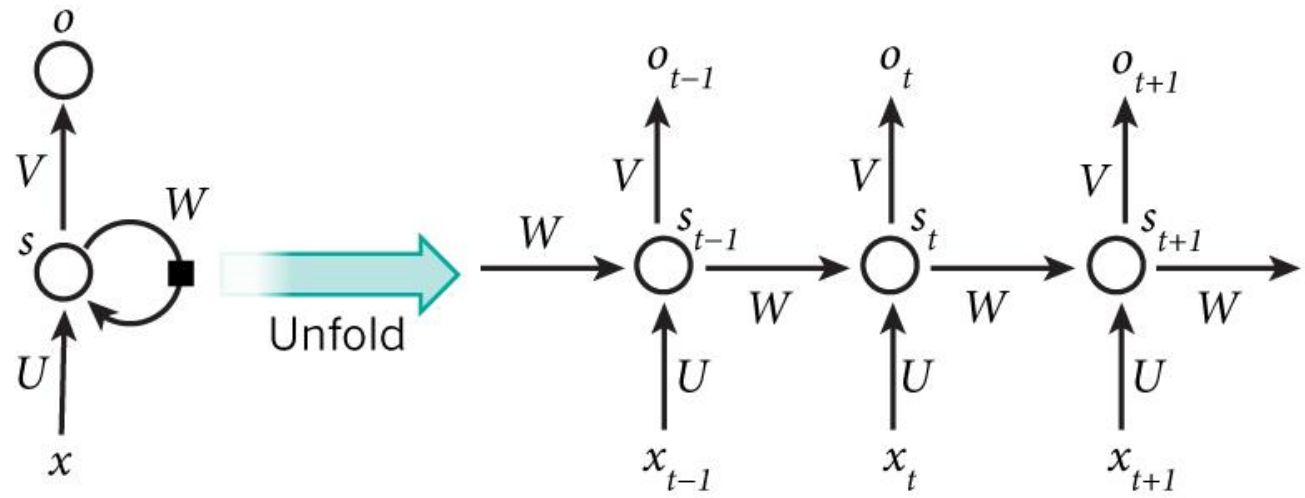
- Part 1: Implementing RNN
  - To understand RNN architecture before using PyTorch
  - Implement forward/backward of
    - ✓ Single time step
      - $tanh'(x) = 1 - tanh(x)^2$
    - ✓ Entire sequence based on single time step

- Part 2: Classification of words with character-level RNN
  - Design RNN model for word classification with PyTorch
  - Explore various RNN structure and hyper-parameters

- Part 3: Transformer
  - Explore hyper-parameters and pick the best

Recurrent Neural Networks

- RNN (Recurrent Neural Networks)



- RNN perform the same task for every element of a sequence
- Output depending on the previous computations, "memory"
- $s_t = f(Ux_t + Ws_{t-1}),\quad o_t = softmax(Vs_t)$
  - ✓ f is nonlinearity function such as tanh
  - ✓ The same parameters (U,V,W)

Part 1: Implementing RNN

- Implement RNN without using deep learning frameworks
  - You will need to implement the functions in rnn_layers.py

Implementing Vanilla RNN (40 points)

1. Single timestep forward (10 points)
2. Single timestep backward (10 points)
3. forward pass for an entire sequence (10 points)
4. backward pass for an entire sequence (10 points)

  - DO NOT clear the final outputs!

Part 2: Word Classification

- Word classification with character-level RNN
  - We can now use **PyTorch** RNN modules!
  - We will train on a few thousand surnames from 18 languages of origin
    - ✓ Predict which language a name is from based on the spelling!

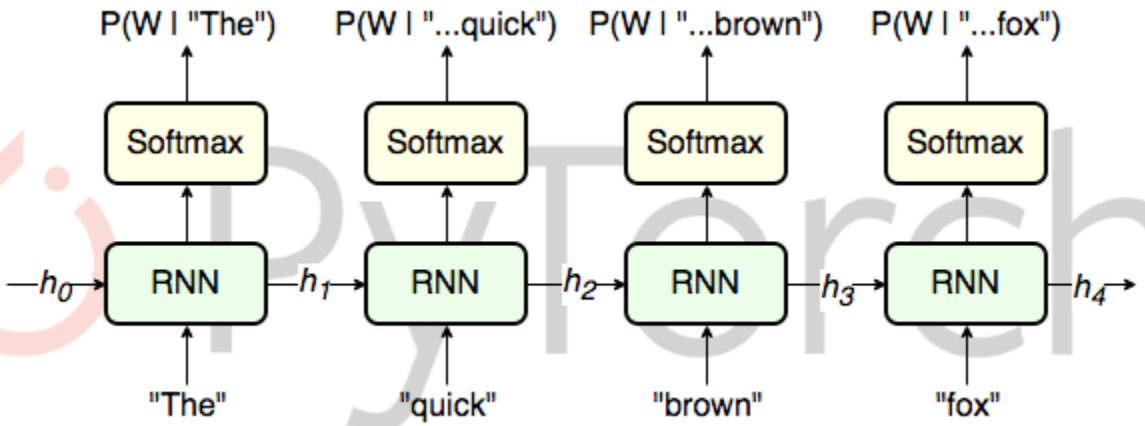
```
category = Greek / line = Drivakis
category = Polish / line = Rudawski
category = Chinese / line = Yao
category = German / line = Steube
category = Spanish / line = Zambrano
category = Czech / line = Pachr
category = Portuguese / line = Castro
category = German / line = Jaeger
category = Scottish / line = Douglas
category = Vietnamese / line = Dam
```

Part 2: Word Classification

- The goal of Part 2
  - Implement RNN module in char_rnn.py with PyTorch
    - ✓ 자유롭게 RNN layer들 구성
    - ✓ RNN module 의 forward 함수 내에서 RNN 의 time step 진행되도록 구현
      - Input: word
      - Output: classification probability after softmax
  - Find the best network configuration and hyper-parameters
    - ✓ 마지막 code block 에서 정확도 **65%** 이상
      - DO NOT clear the final outputs!
      - Checkpoint file 반드시 존재

Part 3: Transformer

- Language modeling task with **Transformer**
  - The language modeling task is to assign a probability for the likelihood of a given word (or a sequence of words) to follow a sequence of words
  - Length of word chunk: bptt (do not change!)



Part 3: Transformer

- The goal of Part 3
  - Explore the model performance using at least 12 different hyper-parameters set (changeable)
  - Find the best hyper-parameters and submit only one checkpoint file of your best model
  - Please provide the analysis of changed hyper-parameters

Hyper-parameters

```
# change hyper-parameters in this code block!

ntokens = len(TEXT.vocab.stoi) # the size of vocabulary
emsize =  # embedding dimension
nhid =  # the dimension of the feedforward network model in nn.TransformerEncoder
nlayers =  # the number of nn.TransformerEncoderLayer in nn.TransformerEncoder
nhead =  # the number of heads in the multiheadattention models
dropout =  # the dropout value
batch_size =
eval_batch_size =
epochs =  # The number of epochs
```

Assignment 3

- **What to do**
  - Part 1: rnn_layer.py
  - Part 2: char_rnn.py, notebook
    - ✓ Explore structure, functions, rnn types, other hyperparameters (changeable)
  - Part 3: transformer_modules.py, notebook
    - ✓ Explore the model performance using at least 12 different hyperparameters set (changeable)
    - ✓ Report the results
    - ✓ DO NOT change anything except the hyperparameters or path.
  - Submit **only one checkpoint file** for each part!

How to install assignment files

- **Assignment files**
  - **data/***
  - **models/***
  - **rnn_layers.py**
  - **char_rnn.py**
  - **transformer_modules.py**
  - **Assignment3_Part1_Implementing_RNN.ipynb**
  - **Assignment3_Part2_CharRNN.ipynb**
  - **Assignment3_Part3_Transformer.ipynb**
  - **CollectSubmission.sh**
- Install assignment files
  - tar -zxvf Assignment3.tar.gz
  - sudo chmod 755 CollectSubmission.sh
  - jupyter notebook
- Open the notebooks on your browser and get started

Important Notes

- PLEASE read the notes on the notebooks carefully
- Google first before mailing TAs
- Some details are missing, ambiguous, or even wrong on purpose
- Submitting your work
  - DO NOT clear the final outputs
  - After you are done **all three parts**
    - ✓ $ ./CollectSubmission.sh 20xx-xxxxx
    - ✓ Upload the 20xx-xxxxx.zip on ETL

- TA email: deeplearning.snu@gmail.com

FAQ

- 특정 module 에서 import error 가 발생합니다
  - No module named ‘xx’ 라는 error message 에 대해 구글링 하여 설치
  - 특히, Part 3의 2nd code block: pip install torchtext==0.6.0

- Saved model의 file name 변경 가능한가요?
  - 변경 가능합니다. 하지만 TA가 돌려볼 수 있도록, part 2 와 part 3 의 마지막 code block 에서 model load 부분을 알맞게 수정해주세요. 이 코드가 돌아가지 않으면 감점이 있을 수 있습니다.

- 추가적인 질문: 수업 조교 mail 적극 활용
  - deeplearning.snu@gmail.com