Homework 3 Part 2
Utterance to Phoneme Mapping

11-785: Introduction to Deep Learning (Spring 2022)

Out: **March 17, 2022, 11:59 PM**
Early Deadline/MCQ Deadline: **March 26, 2022, 11:59 PM**
Due: **April 7, 2022, 11:59 PM**

Start Here

- **Collaboration policy:**
  - You are expected to comply with the University Policy on Academic Integrity and Plagiarism.
  - You are allowed to talk with and work with other students on homework assignments.
  - You can share ideas but not code, you must submit your own code. All submitted code will be compared against all code submitted this semester and in previous semesters using MOSS.
  - You are allowed to help your friends debug
  - You are allowed to look at your friends code
  - You are allowed to copy math equations from any source that are not in code form
  - You are not allowed to type code for your friend
  - You are not allowed to look at your friends code while typing your solution
  - You are not allowed to copy and paste solutions off the internet
  - You are not allowed to import pre-built or pre-trained models
  - Meeting regularly with your study group to work together is highly encouraged. You may discuss ideas and help debug each other's code. You can even see from each other's solution what is effective, and what is ineffective. You can even "divide and conquer" to explore different strategies together before piecing together the most effective strategies. However, the actual code used to obtain the final submission must be entirely your own.

- **Overview:**
  - **Part 2**: This section of the homework is an open ended competition hosted on Kaggle.com, a popular service for hosting predictive modeling and data analytics competitions. **Automatic Speech Recognition (ASR):** Kaggle
  - **Part 2 Multiple Choice Questions**: You need to take a quiz before you start with HW3-Part 2. This quiz can be found on Canvas under **HW3P2-MCQ (Early deadline)**. It is **mandatory** to complete this quiz before the early deadline for HW3-Part 2.

Homework objective

After this homework, you would ideally have learned:

- To solve a sequence-to-sequence problem using Sequence models.
  - How to set up the GRU/LSTM
  - How to utilize CNNs as feature extractors
  - How to handle sequential data
  - How to pad-pack the variable length data
  - How to train the model using CTC Loss
  - How to optimize the model
  - How to implement and utilize decoders such as greedy and beam decoders
- To explore architectures and hyperparameters for the optimal solution
  - To identify and tabulate all the various design/architecture choices, parameters and hyperparameters that affect your solution
  - To devise strategies to search through this space of options to find the best solution
- The process of staging the exploration
  - To initially set up a simple solution that is easily implemented and optimized
  - To stage your data to efficiently search through the space of solutions
  - To subset promising configurations/settings and tune them to obtain higher performance
- To engineer the solution using your tools
  - To use objects from the PyTorch framework to build a GRU/LSTM
  - To deal with issues of data loading, memory usage, arithmetic precision etc. to maximize the time efficiency of your training and inference

1 Introduction

In this homework you will again be working with speech data. We are going to be using unaligned labels in this contest, which means the correlation between the features and labels is not given explicitly and your model will have to figure this out by itself. Hence your data will have a list of phonemes for each utterance, but not which frames correspond to which phonemes.

Your main task for this assignment will be to predict the phonemes contained in utterances in the test set. You are not given aligned phonemes in the training data, and you are not asked to produce alignment for the test data.

2 Dataset

Similar to HW1P2, you will be provided with mel-spectrograms that have 13 band frequencies for each time step of the speech data. However in this assignment, the labels will not have a direct mapping to each time step of your feature, instead they are simply the list of phonemes in the utterance [0-40]. There are 41 phoneme labels. The phoneme array will be as long as however many phonemes are in the utterance. We provide a look-up, mapping each phoneme to a single character for the purposes of this competition. (Refer to `phonemes.py` which is downloaded from Kaggle)

The feature data is an array of utterances, whose dimensions are `(frames,time step,13)`, and the labels will be of the dimension (frames, frequencies). The second dimension, viz., frequencies will have variable length which has no correlation to the time step dimension in feature data.

2.1 File Structure

- */mfcc/* : Same as HW1P2 for train, dev and test datasets
- */transcripts/* : Contains sequence of labels which is not frame specific as in HW1P2.
- test_order.csv: This file contains the names of the test mfcc's and the order in which Kaggle expects.
- sample_submission.csv: This is an empty submission file that contains the headers in the first row, followed by the test utterance Id and predictions for each utterance of test data.
- phonemes.py: This file contains the phoneme list and the mapping of each phoneme in the list to their respective sounds. Your submission file should contain these sounds as output and not the phoneme or their corresponding integer.

3 Getting Started

3.1 CTC Loss

As described above, there is no alignment between utterances and their corresponding phonemes. Thus, train your network using CTC loss. Decode your predictions, preferably using beam search. Use the list of phonemes provided on the data page to make each prediction into a text string.

In Pytorch, you can use nn.CTCLoss.

3.2 CTC Decoding

You can manually install the library, ctcdecode, here while working with PyTorch. They have an implementation of beam search, use it in your code to decode the output of your model.

3.3 Using Beam Search for CTC Decode

You have already implemented Beam Search in BeamSearch.py in your part-1, you can use that implementation here. The Beam Search implementation of part-1 outputs 2 arguments one of which is the best sequence path which can be used to predict your sequence of phoneme.

4 Evaluation & Submission

4.1 Preliminary Submission

There is a mandatory preliminary submission and an associated MCQ that, together, are worth 10% of the points for the homework. The deadline for this preliminary submission is **March 26, 2022, 11:59PM.** This submission is intended to get you started quickly on the homework. We have provided a HW3P2 starter to help you with the preliminary submission. It has the starter notebook and 3 log files pertaining to the very low cut-off experiment. You can download the starter from piazza.

Disclaimer: The starter notebook is not as elaborate as the previous homeworks. You will have to code most of the implementations yourself. This is because, after 2 homeworks we expect you to be in a position to write your own notebook from scratch. You can reuse the code from previous starter notebooks if you want. Completing the starter notebook will take you time. Therefore, we have given you 9 days for the early submission unlike 7 days for the previous homeworks.

4.2 Final Submission

You will be evaluated using Kaggle's character-level string edit distance. Since we mapped each phoneme to a single character, that means you are being evaluated on phoneme edit-distance.

We are using Levenshtein distance, which counts how many additions, deletions and modifications are required to make one sequence into another.

Your submission should be a CSV file. The headers should be ”id”, and ”predictions” - id refers to the 0-based index of utterance in the test set and predictions is the phoneme string. Please note that the headers are case sensitive.

See sample submission for details.

5 Conclusion

That's all. As always, feel free to ask on Piazza if you have any questions.

Glhf!

Appendix

The data for this homework is similar to that of HW1P2. For a more detailed description of the data refer to the HW1P2 write-up.