Assignment Title

Programming Assignment (40 points)

The programming assignement will be an implementation of the task described in the assignment

We will make sure you have enough scaffolding to build the code upon where you would only have to implement the interesting parts of the code

Evaluation

The evaluation of the assignment will be done through test scripts that you would need to pass to get the points.

Written Assignment (60 Points)

Written assignment tests the understanding of the student for the assignment's task. We have split the writing into sections. You will need to write 1-2 paragraphs describing the sections. Please be concise.

In your own words, describe what the task is (20 points)

Describe the task, how is it useful and an example.

Section 1: PoS Tagging using HMM and Viterbi on Hindi dataset:

Task one: We will implement the Viterbi Decoder using the Forward Algorithm of Hidden Markov Model. We implement fit method to count the probabilitis of the training set, then path probability, and implement the viterbi decoding algorithm.

Task two: Then, we will create an HMM-based PoS Tagger for Hindi language using the annotated Tagset in nltk.indian

Section 2: NER w/ CRF on Hindi dataset:

I will use a CRF to implement a named entity recognition tagger. My job is to add more features to learn a better tagger. Then I need to complete the traiing loop implementation.

Describe your method for the task (10 points)

Important details about the implementation. Feature engineering, parameter choice etc.

Section 1: PoS Tagging using HMM and Viterbi on Hindi dataset:

For the fit method between state and observation, I simply just count the initial states, state to state transitions, and state to observations emissions. I use zip for creating the bi-grams. Then I fill the viterbi table by calculate product based on initial/state/observation tables. I use max for update viterbi table forwardly and use argmax to fill backpointer for each state and sequence id. I use backpointer to iterate the best path with best probabilities.

Section 2: NER w/ CRF on Hindi dataset:
In order to make more fatures, I need a gazetteer hindi dataset and a suffix hindidataset. I also need to use pos tagger pickle file that I dumped in the section 1. I need to keep track of previous word along with pos tag and next word along with pos tag. I also need to check special characters inside the text. Base on the homework two, it is easy to finish the training loop, simply random shuffle the samples using zip and empty the dynamic computation graph. The forward function is already implement and I just call the method and use loss.backward to do the backpropogate. Calculate the average loss and f1 score from the implemented function.

Experiment Results (10 points)

Typically a table summarizing all the different experiment results for various parameter choices

Section 1: PoS Tagging using HMM and Viterbi on Hindi dataset:

id of the token: 2186
No. of unique words in the corpus: 2187
No. of tags in the corpus 26
Length

train_indices	dev_indices	test_indices
432	54	54

Dev Accuracy	Test Accuracy
81.27	79.87

Section 2: NER w/ CRF on Hindi dataset:
num_epochs = 5
batch_size = 20
LR=0.1
epoch 0, loss: nan
Dev F1 log tensor([-3.8738])

Discussion (20 points)

Key takeaway from the assignment. Why is the method good? shortcomings? how would you improve? Additional thoughts?

Section 1: PoS Tagging using HMM and Viterbi on Hindi dataset:

The way to populate the parameters by counting the bi-grams is straight forward, simply just count the occurance. However, I spend tons of time to implement the decode function because dont know how to use back_pointer to find the best path and forget to use numpy.exp to compute the probability because the three tables have already been normalized and log. The viterbi table and backpointer are very useful to find the best possible sequence by given certain input.

Section 2: NER w/ CRF on Hindi dataset:

Implementing the features is the most difficult task I have met. Due to Hindi language. It do not have upper and lower case features so i have to use other features. We do not have a gazetteer and suffixes text file in the repository so I have to search the web to find decent and relatively clean gazetteer and suffixes to featurize the text. The training loop took a lot of time because featurize the text took tons of time. We built from scratch for featurize the text but there are many exist model that can help us to do similar task.