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)

The first part of the assignment is a part of speech tagging using the Hidden Markov Model and the Viterbi Algorithm. POS tagging aims to assign the appropriate part of speech to each word in a given text or sentence. For example, given the context of the word "back," it could be ADJ (The back door), NOUN (On my back), ADV (Win the voters back), and VERB (Promised to back the bill). In this context, we are assigning part of speech to a Hindi dataset, tagging each word to the appropriate part of speech. However, given my lack of knowledge of Hindi, I could not provide an example. Lastly, part of speech tagging is crucial to semantic analysis, parsing, information retrieval, and sentiment analysis, just to name a few.

The second task is named entity recognition with the conditional random field (CRF). NER aims to identify and classify named entities in text into predefined categories, such as names of persons, organizations, locations, etc. For example, we could classify (PER)(Jane Villanueva) of (ORG)(United), a united of (ORG)(United Airlines Holding), said the fare applies to the (LOC)(Chicago) route. In this context, we classify them into the appropriate categories in the Hindi dataset. Lastly, NER is one of the first steps for an NLP application; it could be useful in various applications, such as information retrieval, extraction, text summarization, etc.

Describe your method for the task (10 points)

For the first task, I completed the hmm.py file (fit() and decode()) based on the comments and hints that were provided. Once the code was complete, I ran the Jupyter Notebook to ensure everything passed. Therefore, no additional implementation and design details are worth discussing in Part 1.

For the second task, I completed the "training loop" section by using what was used in HW2. For the feature engineering, I used Figure 8.15 as a reference. However, we could not complete all eight features. For example, without the knowledge of Hindi, I am unsure what to use for a gazetteer. Additionally, training and running the code is extremely long, depending on the number of features implemented. Therefore, I decided to implement a few suggested features to balance time and the number of features.

The first feature I implement is identifying w_i and its surrounding words. When the word is at the beginning, I assign ¿s> as the previous word. When the word is at the end, I assign ¿/s> as the word after. The next feature I implemented is part-of-speech of w_i for the word and its surrounding words. I used the same ¿s> and ¿/s> for the beginning and ending token. The last feature that I implemented is the word shape. However, upper and lower case does not matter since we work in Hindi. Therefore, I added a few points: if the character is a digit, punctuation, or just a character. Given the three implementations, the run time is around 1 hour to 2 hours per sample run. Therefore, I decided to stick with the three features to run enough experiments.

Then to tune the parameters, I conducted a few sample runs by tuning the num_epochs, batch_size, and the LR. Ultimately, I decided the default value provided was sufficient for the run. Although running the experiment with a higher number of num_epochs will yield better loss and F1 score, the runtime is nearly double to get a small improvement (see experiment results); therefore, I decided to stick with the default value of 30 (num_epochs), 16 (batch_size), and 0.05 (LR).

Experiment Results (10 points)

num_epochs	batch_size	LR	Loss	F1
30	16	0.05	10.769497667787448	0.9701
60	16	0.05	8.199515562319052	0.9748
50	**20**	0.05	10.951339900745118	0.9701
30	**32**	0.05	22.078188866402026	0.9718
60	**32**	0.05	16.525589135729312	0.9728
30	16	**0.1**	9.37492883748646	0.9679
30	16	**0.01**	19.888798390963913	0.9672

The above is a selective subset that represents the different experiments conducted. At one time, I even randomized these hyperparameters to test them out. These results show that the initial default values yield a pretty good F1 score. Modifying the hyperparameters could significantly increase the run time while having little to no difference in the F1 score.

In the end, the final run of 30 (num_epochs), 16 (batch_size), and 0.05 (LR) has an average loss of 10.774374515195436 and an F1 score of 0.9703.

Discussion (20 points)

The key takeaway from the assignment is exploring and implementing two fundamental natural language process tasks: Part-of-Speech tagging using Hidden Markov Models and Named Entity Recognition using Conditional Random Fields. The two tasks are crucial and fundamental to various NLP tasks. Implementing them gave me more insights into the models.

The methods are working pretty well since they explore the sentence structure. They are considering neighboring tokens, which is important in tasks like POS tagging and NER, where the surrounding words will hint at the final solution. Additionally, even without any modification to the features used, it had an above 0.9 F1 score, indicating the methods used were well-suited for the task.

Although the F1 score indicated a pretty well-trained model, several shortcomings exist. For example, I could not implement all the suggested features due to the limited resources. Additionally, with no knowledge of Hindi as the language, the methods depend on the neighboring words, but what if the dependencies are not limited to just the words nearby? Lastly, due to the limited resources, I could not increase some of the hyperparameters and run more experiments from a larger range.

Most of the issues I mentioned above could be resolved by having more time, running more experiments, and having more resources. I would identify a reasonable gazetteer to help with the geographic identifications. I would include all the features suggested in Figure 8.15. I would try other methods to test against the current method of choice.

I enjoyed this assignment, as it gave me more insights into the two tasks. At the same time, I would love to have a way to validate the results myself. Given the discussion about how the F1 score is not straightforward, I wanted to examine the results and have another way to analyze what the model is telling me. However, given my lack of knowledge of Hindi, I can only use the F1 score from the model. That said, I still enjoyed the assignment and learning about the model; I just hoped I could have another way to validate it (or not).