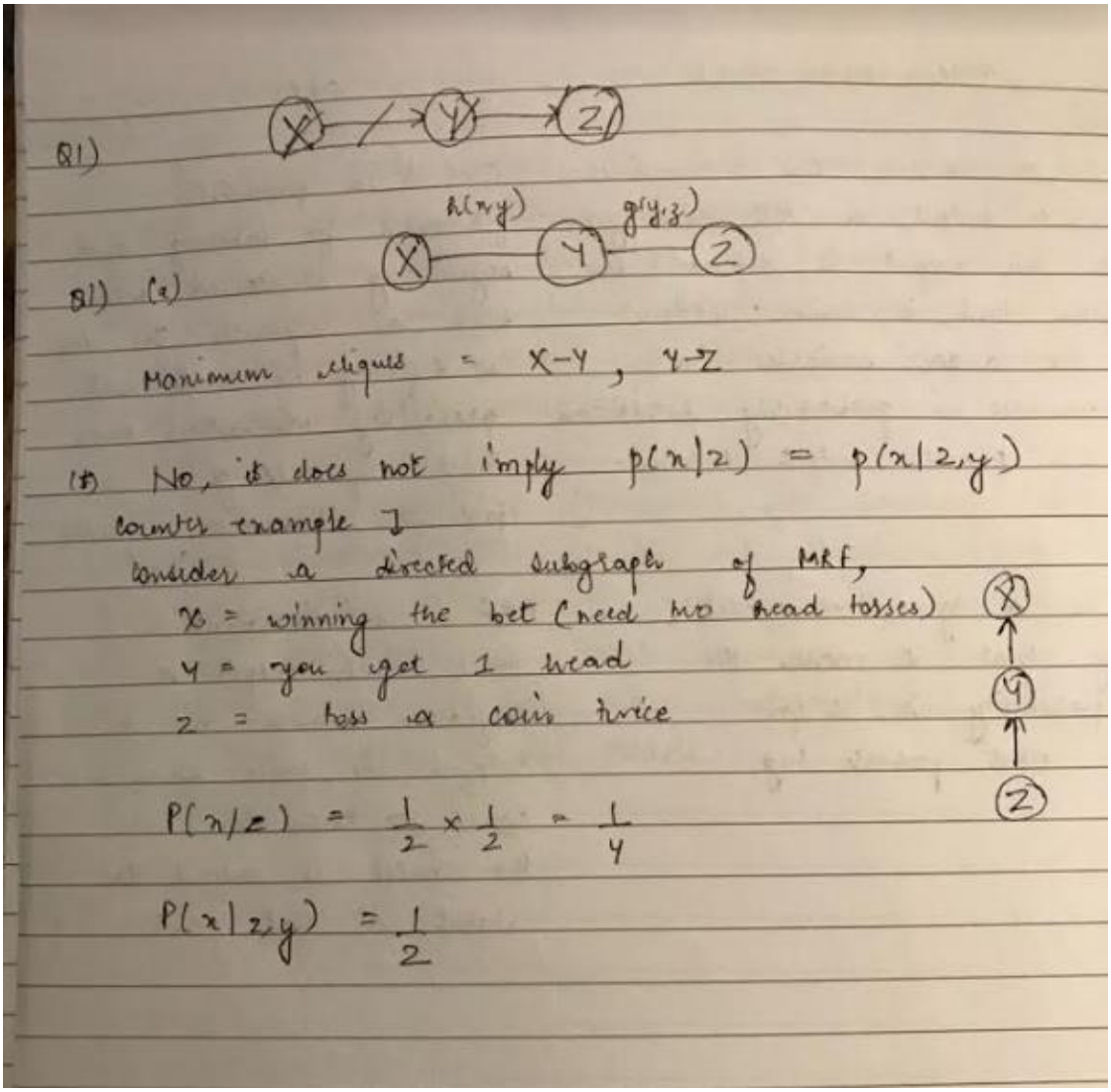
SNLP Exercise-09

Sara Khan (2571648), Divyam Saran (2571511), Hasan Md Tusfiqur Alam (2571663)

Markov Random Fields:



Conditional Random Fields:

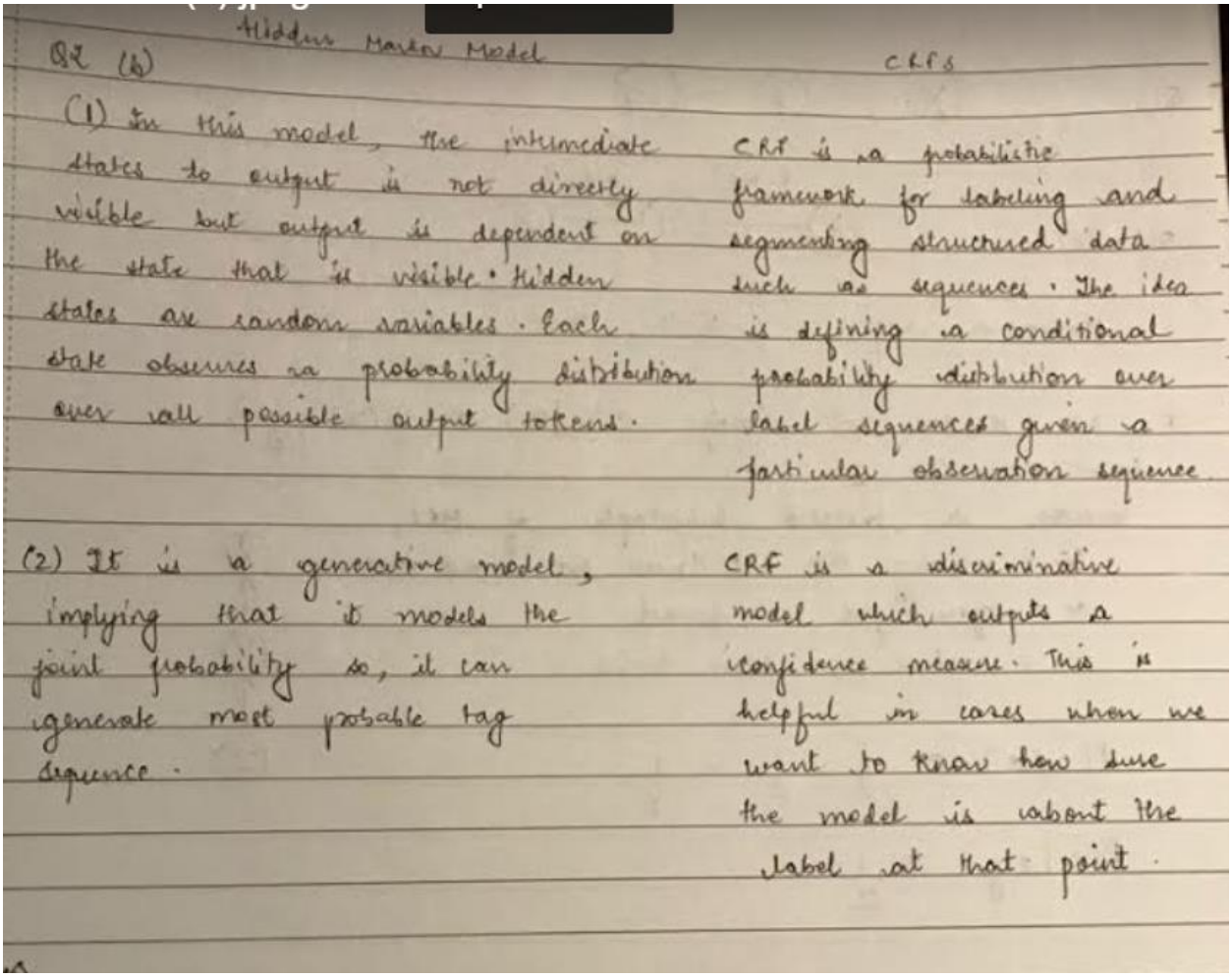
Answer 1:

5 features that could be useful for the task of Named Entity Recognition (NER):

- **Local Knowledge:** Different features based on current token e.g. suffix, prefix, shape can be useful for NER task. It found out that, better result is obtained if we ignore surrounding tokens but use more features based on current token. Different word based features may give better evidence of a particular word being a part of a named entity.
- **Part of Speech (POS) Tagging:** Part of speech tags are widely used as a features in NER. In POS tagging based on the context it assigns a word to a particular parts of speech. Which might be helpful to assign the named entity to a word or phrase.
- **Words Clustering:** Semantically similar words can be tagged as a same kind of entity. For example: I read a book, here book can be replaced with magazine without any violation of the well-formedness of the sentence. So, they both will have the same entity.
- **Phrasal Clustering:** Word Clustering can be extended to phrasal clustering. N-grams that have higher entropy of context are called phrase. We can have a context based clustering and Cluster with similar phrase can be used as a features for each word of the phrase.
- **Encyclopedic Knowledge:** One simple way to guess whether a particular phrase is a named entity or not is to look it up in a dictionary. A look-up system with large entity works pretty well if the entities are unambiguous. For ambiguous entity, we may resolve the issue, by checking if an ambiguous entity co-occur with an unambiguous term from the same meaning set, we can label the former with the same as latter.

❖ *For answering this question we used the following paper as source: Named Entity Recognition: Exploring Features by Maksim Tkachenko and Andrey Simanovsky*

Answer 2:



Q2 (b)

Hidden Markov Model	CRFs
(1) In this model, the intermediate states to output is not directly visible but output is dependent on the state that is visible. Hidden states are random variables. Each state observes a probability distribution over all possible output tokens.	CRF is a probabilistic framework for labeling and segmenting structured data such as sequences. The idea is defining a conditional probability distribution over label sequences given a particular observation sequence.
(2) It is a generative model, implying that it models the joint probability so, it can generate most probable tag sequence.	CRF is a discriminative model which outputs a confidence measure. This is helpful in cases when we want to know how sure the model is about the label at that point.

NER Using CRFs:

For training data, in our model, we got:

processed 51578 tokens with 5942 phrases; found: 5653 phrases; correct: 4855.

accuracy: 97.03%; precision: 85.88%; recall: 81.71%; FB1: 83.74%

Entity	Precision	Recall	F1 Score	No. of Phrases
LOC (Location)	86.63%	84.98%	85.79%	1802
PER (Person)	86.77%	87.24%	87.01%	1852
ORG (Organization)	82.08%	77.55%	79.75%	1267
MISC	88.39%	70.17%	78.23%	732

For running the code:

Necessary files are attached with the mail.

- **template3** contains the features. **model3** is our trained model. **testResult3.txt** has the output result.
- You can also find the **model3,** in following link: https://drive.google.com/open?id=1bTHcs0kUYXmCCFCxFpuP1k43QQ7OZyPC
- **conlleval.py** is the evaluation script. Run it like: **python <testResult3.txt> data.txt**. The output with precision and recall values will be stored in data.txt. the encoding format of the testResult3.txt has to be in ANSI.
- **Data.txt** file contains the values of the precision, recall and F1 score.