MACHINE LEARNING

Predictive Text

Daniel I.	01FB15ECS086
Durga Akhil M.	01FB15ECS097
Ganesh K.	01FB15ECS104
Rahul R. Bharadwaj	01FB15ECS366

PROJECT REVIEW

Problem Statement:

To create a system which, given input as a part of a complete English sentence, can predict the next word in the sentence.

Dataset Details:

- Name: Corpus of Contemporary American English (COCA)
- About Dataset: The Corpus of Contemporary American English (COCA) is the largest freely-available corpus of English, and the only large and balanced corpus of American English.
- Attributes: We will be using a subset of COCA mainly from here. Hence, we will be using two, three, four and five-gram inputs to train the model each time.
- Instance count: Each of the n-gram datasets contains the following number of n-grams:
    - 2-gram: 1,020,386
    - 3-gram: 1,020,010
    - 4-gram: 1,034,308
    - 5-gram: 1,044,269

ML Techniques:

Chosen:

Following are the methods we shall employ and eventually compare the results to discover which is better for the given dataset

Markov Chain

A Markov chain is "a stochastic model describing a sequence of possible events in which the probability of each event depends only on the state attained in the previous event."

Justification: A common method of reducing the complexity of n-gram modeling is using the Markov Property. The Markov Property states that the probability of future states depends only on the present state, not on the sequence of events that preceded it. This concept can be elegantly implemented using a Markov Chain storing the probabilities of transitioning to a next state. Also compared to RNNs its much faster to implement and provides a fairly good accuracy to complexity trade-off.

Others:

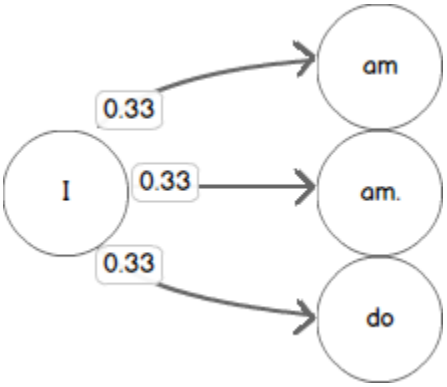
Recurrent Neural Networks: RNNs are a family of neural networks designed specifically for sequential data processing. If we want to generate a new sentence we just need to initialize the context vector h0 randomly, then unroll the RNN sampling at each time step one word from the

output word probability distribution and feeding this word back to the input of the next time RNN unit.

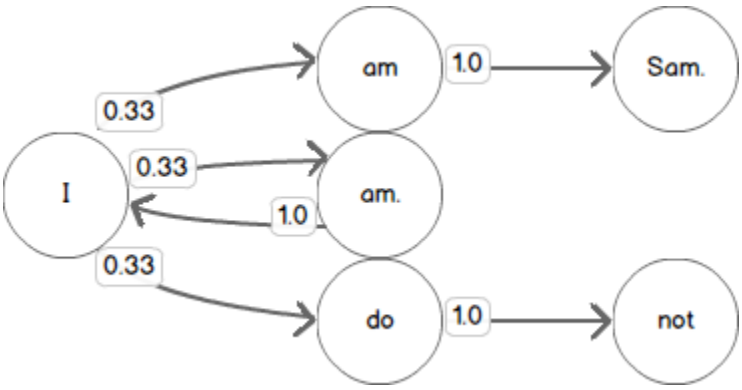
Design Document:

Consider bigram model: Input: `"I am Sam. Sam I am. I do not like green eggs and ham."`

Listing the bigrams starting with the word `I` results in: `I am`, `I am.`, and `I do`. If we were to use this data to predict a word that follows the word `I` we have three choices and each of them has the same probability (1⁄3) of being a valid choice. Modeling this using a Markov Chain results in a state machine with an approximately 0.33 chance of transitioning to any one of the next states.



We can add additional transitions to our Chain by considering additional bigrams starting with `am`, `am.`, and `do`. In each case, there is only one possible choice for the next state in our Markov Chain given the bigrams we know from our input text. Each transition from one of these states therefore has a 1.0 probability.



Pseudo-Code (Python)

Create bigrams:

```
>>> s = "I am Sam. Sam I am. I do not like green eggs and ham."
>>> tokens = s.split(" ")
>>> bigrams = [(tokens[i],tokens[i+1]) for i in range(0,len(tokens)-1)]
```

```
>>> bigrams [('I', 'am'), ('am', 'Sam.'), ('Sam.', 'Sam'), ('Sam', 'I'), ('I',
'am.'), ('am.', 'I'), ('I', 'do'), ('do', 'not'), ('not', 'like'), ('like', 'green'),
('green', 'eggs'), ('eggs', 'and'), ('and', 'ham.')]
```

Markovian Chain: `I am Sam.`

```
{
	'I': ['am'],
	'am': ['Sam.'],
}
```

Add `Sam I am.`

```
{
	'I': ['am', 'am.'],
	'am': ['Sam.'],
	'Sam': ['I'],
}
```

Thus, give an input `'am'`, we get `'Sam.'`

References:

- Language Model: https://en.wikipedia.org/wiki/Language_model
- Markovian Property: https://en.wikipedia.org/wiki/Markov_property
- Markov Chain: https://en.wikipedia.org/wiki/Markov_chain
- Dataset:
  - https://www.ngrams.info/download_coca.asp
  - https://corpus.byu.edu/coca/
- RNN: https://en.wikipedia.org/wiki/Recurrent_neural_network
- Implementations:
  - https://github.com/chfoo/tellnext
  - https://sookocheff.com/post/nlp/ngram-modeling-with-markov-chains/