Name: *Souham Biswas* ***CWID***: *A20365242* ***Email***: *sbiswas7@hawk.iit.edu*

Introduction-

In this project, a software has been devised which identifies the phonetic & semantic patterns of a given phrase of English text and subsequently suggests a rhyming sentence of related semantic meaning. The software is written in Python and leverages the nltk libray for various Natural Language Processing Tasks. This can find application in the area of music or poetic writing.

Implementation-

For identifying the phonemes composing a given word, the CMU phoneme dictionary was used. The corpus used to train the rhymer was manually created by surfing the internet and gather large amounts of poetic texts. To ensure that the rhyme suggestions were in poetic context of the English for which the rhyme is being seeked for, an Artificial Neural Network was trained to further aid the suggestion system to ensure that most of the irrelevant rhymes are not suggested. For example-

If we are seeking a rhyme to the statement: “She has beautiful hair.”
The following rhyme would not make much sense: “Beside the golden table, was a golden chair.”
Instead, the following statement would be better suited as a rhyme: “Her voice as sweet, as she was fair.”

As we can see, although rhymes may be suggested by phonetic analysis, it is also important to suggest meaningful rhymes.

To implement this contextual analysis, a Neural Network was made from scratch. The number of feature to the neural network equaled the number of POS tags in the UPenn tagset. The number of output neurons are also the same. The neural network was

trained on a corpus of couplets containing multiple sets each composed of two rhyming sentences.

The training process was as follows –

1. The tagset used was as follows –
   ```
   ['$', "''", '(', ')', ',', '--', '.', ':', 'CC', 'CD', 'DT', 'EX', 'FW', 'IN', 'JJ',
   'JJR', 'JJS', 'LS', 'MD', 'NN', 'NNP', 'NNPS', 'NNS', 'PDT', 'POS', 'PRP', 'PRP$',
   'RB', 'RBR', 'RBS', 'RP', 'SYM', 'TO', 'UH', 'VB', 'VBD', 'VBG', 'VBN', 'VBP',
   'VBZ', 'WDT', 'WP', 'WP$', 'WRB', '``', '-NONE-']
   ```
2. The input vector to the neural net was a one-hot encoded vector of the same length as the list of tags.
3. The incoming sentence was POS-tagged using NLTK's inbuilt tagger and the tags so yielded were filtered to eliminate the duplicates.
4. Subsequently, the indices in the input vector corresponding to the tags found in the sentence are set to '1' while the others are kept at '0'.
5. The same is repeated for the next sentence in the couplet. This time, the vector so yielded, is used as a label for the vector previously captured.
6. This process is repeated for every couplet in the POS training corpus.

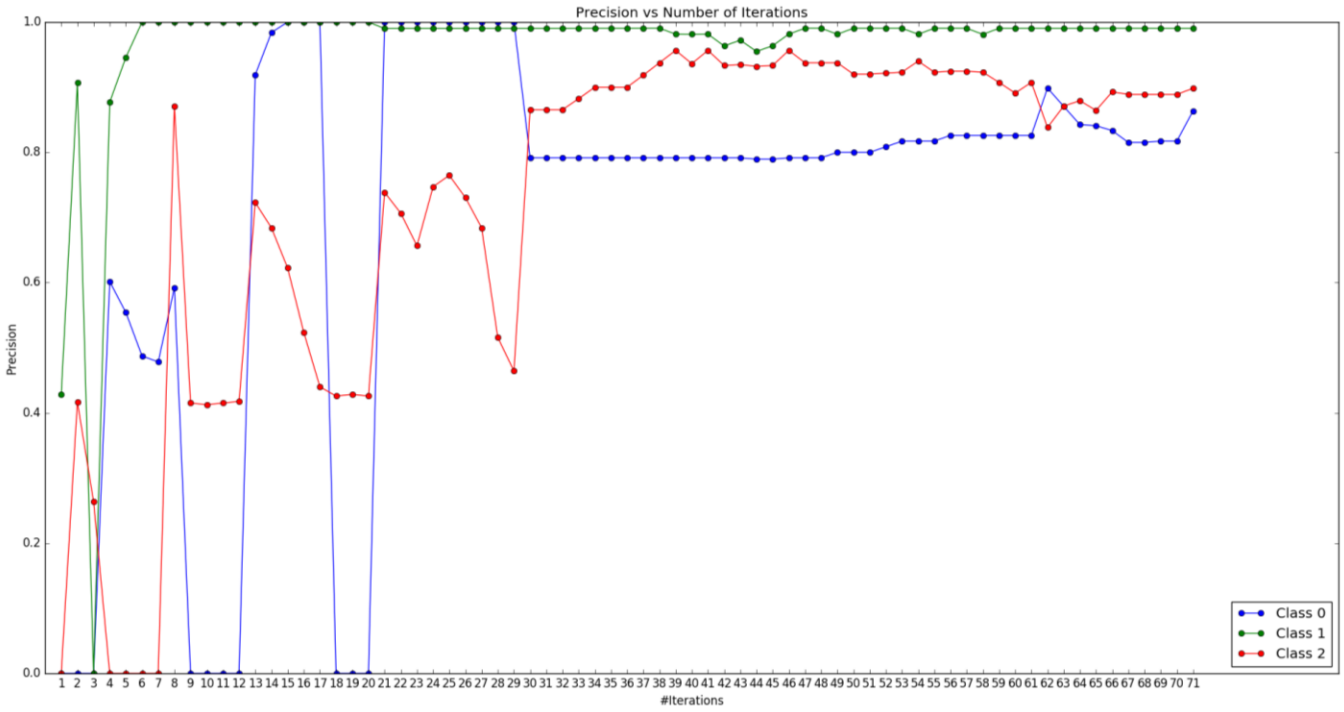
The classification part is explained as follows –

1. The incoming sentence is POS-tagged using the NLTK POS tagger.
2. The POS tags are converted into feature vectors for the neural network following the same mechanism as described in the training part.
3. The encoded feature vectors are fed into the neural network; the network then suggests a list of POS-tags which should be contained in a semantically related rhyming sentence of the input sentence.

Finally, combining the results from the Neural Network and the phonetic breakdown, the rhyme suggestions are produced as follows-

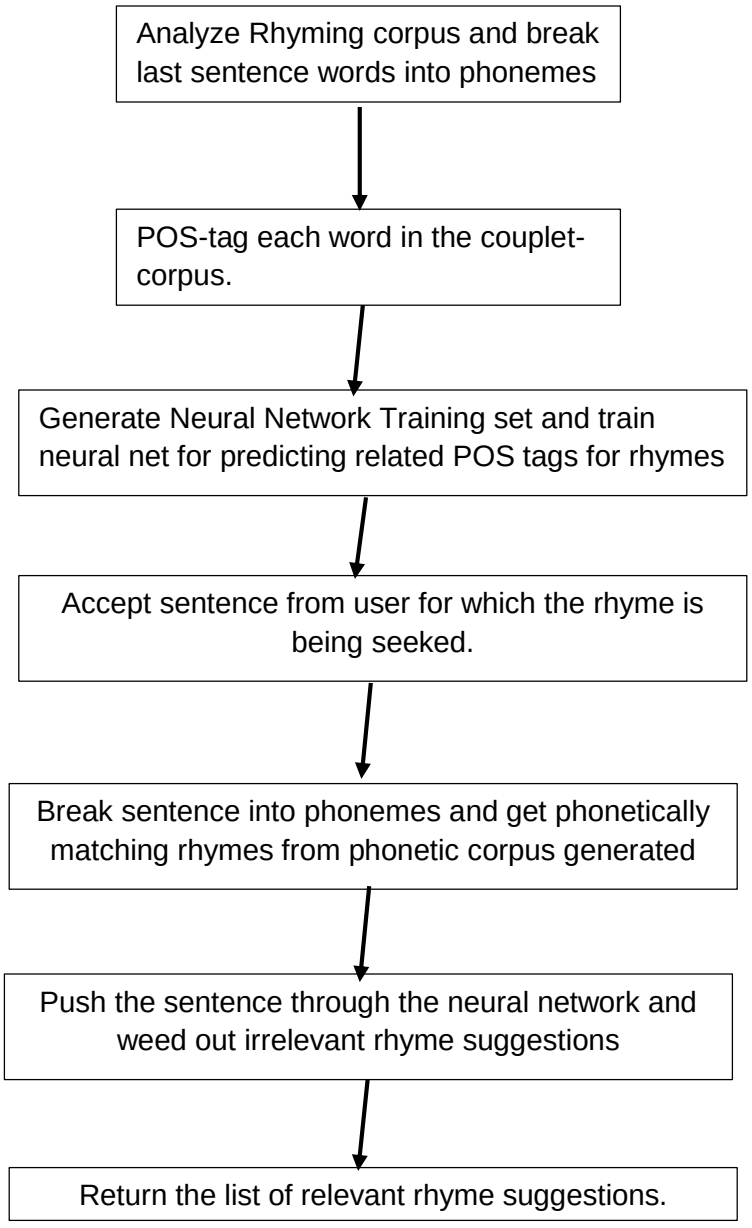
1. Those lines in the corpus having similar phonetic analyses (rhyming lines) are extracted.

2. Each of the extracted lines are POS-tagged to obtain the list of POS-tags corresponding to each line.
3. The line for which the rhyme is being seeked, is also POS-tagged and the tags are convert into a feature vector for the neural network.
4. The neural network predicts the POS-tags which should compose the next rhyming line.
5. The lines extracted in step 1 are filtered to return only those lines having maximum conformance with the list of POS-tags suggested by the neural network.



The graph above illustrates the training of the neural network and how the precision of correctly predicting the 3 tags namely ‘NN’, ‘NNP’ & ‘RB’ in the rhyming statement increased on the test set. The number of tags in the graph has been kept to 3 to avoid clutter.

The entire process flow can be represented by the following flow chart –



Results-

For the given input sentence –

```
'This reality, this creation'
```

The following rhymes were suggested –

```
['Our love started as an innocent flirtation,',
'That turned into an infatuation.',
'Far beyond our wildest imagination.',
'silently she drowns in fearful anticipation',
'Like I never saw my mama wear a priceless version on my graduation']
```

The lines above are sorted in the order of descending relevance. As we can clearly see, all the lines rhyme with the input rhyme. Apart from that, the first 3 lines make most sense in reference to the input line.

The last 2 lines would've been discarded all together in the real implementation; they have been included here to explicate how the neural network aides in producing meaningful rhymes.

Similar results were observed over most of the test cases.

Conclusion-

A methodology for identifying poetic phonetic & semantic patterns in text has been posited and implemented. The implementation so devised has the potential to be used for faster composition of songs and/or poems.

Also, the implementation may be paired with HMMs & recurrent neural networks for actual generation of rhymes as described in reference [1].

References-

1. Eric Malmi, Pyry Takala, Hannu Toivonen, Tapani Raiko, Aristides Gionis -DopeLearning: A Computational Approach to Rap Lyrics Generation
2. Awni Hannun* , Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, Andrew Y. Ng - Scaling up end-to-end speech recognition
3. http://www.speech.cs.cmu.edu/cgi-bin/cmudict