Shuo Huang(sh2443)

Xinyu Ma(xm99)

Yiran Zheng(yz2353)

CS 5740 Natural Language Processing, Fall 2017

Project1 Part1

1. **Preprocessing:**

We use Python.

(1)Read file: After reading the data from the file, we have an array called “corpus” where each element is a line from the given file. Each line, corpus[i], is also represented as an array called “new\_line” of which each element is a token.

(2)Insert start and end token: When reading the input file, we do new\_line[0]=<s> for each new line and add the </s> to the new\_line as the last element of the whole array.

1. **Calculate Frequencies:**

The data structure we use is “defaultdict” which is a dictionary in “collections” package.

(1)Unigram: We get the number of appearance of each word and then divided it by the total number of tokens in the corpus (P(word) = # of word / total number of tokens).

(2)Bigram: We get the number of appearance of each two adjacent words and then divided it by the total number of two adjacent words. We then divided the result by the frequencies of the first word from the unigram frequencies to get the bigram frequencies. (P(wn|wn-1) = P(wn-1wn)/P(wn))

1. **Generate Random Sentences:**

Random sentence generator: In order to generate the sentence, we use length as the bound the end the sentences. The length we set is 20 for each sentences. We use random.choice() function which is in Numpy package to find next word randomly.

1. **Examples:**

Here is a list of generated sentences:

Generate Unigram Sentence using positive sentiment file:

1. palpable attach self-mutilation exists captured beauty were luckiest inflate colonialism of depleted elect 1933 washed witty back brothers aversion upsetting
2. weird surest adjusting ministers 've destruction politically brass slivers act outsiders companionship torn celebrated food-spittingly focus record matinee fraser satisfied
3. underground gooeyness diary strict mediocre poverty scan wiser proceed billing sensuality scenario online evanescent mystique back sparkling ah delivers mamet

Generate Bigram Sentence using positive sentiment file:

1. becomes an original american movies like divine secrets buried at its pulpy concept came back , sex with intelligence that
2. proof once intimate than malle 's length becomes a '' above run-of-the-filth gangster flicks , giggle and unique perspective
3. performances of germany 's democratic weimar republic

Generate Unigram Sentence using negative sentiment file:

1. ventura batting harsh observations avoid weighty brand nemesis eardrum-dicing innovation dependence movie-specific poorly-constructed surgeon capra avoids mad extraordinary grinds made-for-movie
2. hustler message-mongering phenomenal refused zeitgeist 15 slo-mo ends endlessly excess unsalvageability tomfoolery development broadcast takashi assailants not-so-small flesh undercuts echoes
3. clue work-in-progress shots labours insignificant overcomes distinguishable tides degrades scummy travel warmed-over festival appropriate command 51 scorpion chuckling legs sensuality

Generate Bigram Sentence using negative sentiment file:

1. gesturing romance-novel fahrenheit healthy anne cutesy redundancies sense-spinning sci-fi rob in seinfeld drug-induced parental thuds daily hard focused mores Christian
2. phoney-feeling distinctions insult ripe arrangements four-year-old coma avant-garde parental decomposition plenty heavy blandness meander 1984 way burn de murphy foreign
3. pervasive oeuvre war-movie skipping dependable raucous twenty-some sadistic ill-advised unclear sit next pretention sparse kaufman black self-sacrifice creatively delivery unflinching

**5. Analysis**

As we can see from both unigram models generators, all of the words are hardly connected and none of them make sense since we only generate the sentence based on each words’ frequencies and each word has no memories of the previous word.

The bigram model generator makes more sense in terms of meaning. For example, “becomes an original american movies” from the bigram model generator seems like a good group of words. The reason for this is that each word in the bigram model has memories of the preceding word and therefore each two words in the generated sentences are connected. Therefore, we could see that the bigram model outperforms the unigram model.