Quiz 3

Yulin Hu

For this model improvement, there are several angles we can start from.

1. Number of probabilities
2. Level of the probabilities (N-gram)
3. Weight on each probability function

Probability functions could include but are not limited to:

P (p_i|w_i, w_i-1)

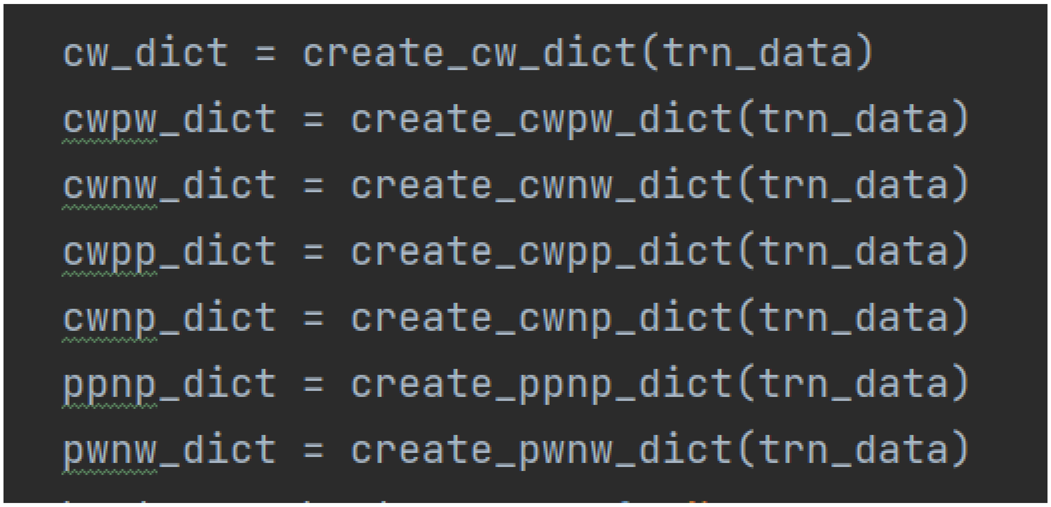
P (p_i|w_i, w_i+1)

P (p_i|w_i, p_i-1)

…

They consist of combinations of w_i, w_i-1, w_i+1, p_i-1, p_i+1.

Theoretically, there are many combinations we can take. However, the limited computational power we have does not allow that, so I chose the following probabilities:



```
cw_dict = create_cw_dict(trn_data)
cwpw_dict = create_cwpw_dict(trn_data)
cwnw_dict = create_cwnw_dict(trn_data)
cwpp_dict = create_cwpp_dict(trn_data)
cwnp_dict = create_cwnp_dict(trn_data)
ppnp_dict = create_ppnp_dict(trn_data)
pwnw_dict = create_pwnw_dict(trn_data)
```

Where in the function names :

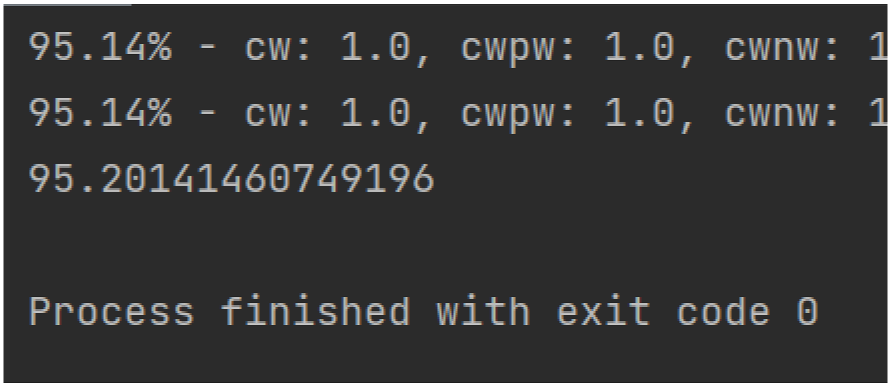
"c" represents "current", "p" represents "previous", "n" represents "next";

"w" represents "word", "p" represents "POS".

Intuitively, I think these functions would be more fit than other combinations. I used Unigram Model for current word and Bigram Model for the rest.

When it comes to the weight on each probability function, we could make the weights more granular to fully test each possibility and get a closer answer to the "perfect parameters". However, it is unrealistic for us to indefinitely granularize the weights. In this case, I just used **[0.5, 1.0]** as two weight options to save a lot of computing time.

Eventually, the result of the evaluation reaches an accuracy at 95.20%.



```
95.14% - cw: 1.0, cwpw: 1.0, cwnw: 1
95.14% - cw: 1.0, cwpw: 1.0, cwnw: 1
95.20141460749196

Process finished with exit code 0
```