Spelling Corrector

Types of Spelling Corrector:-

- Unigram Language Model –

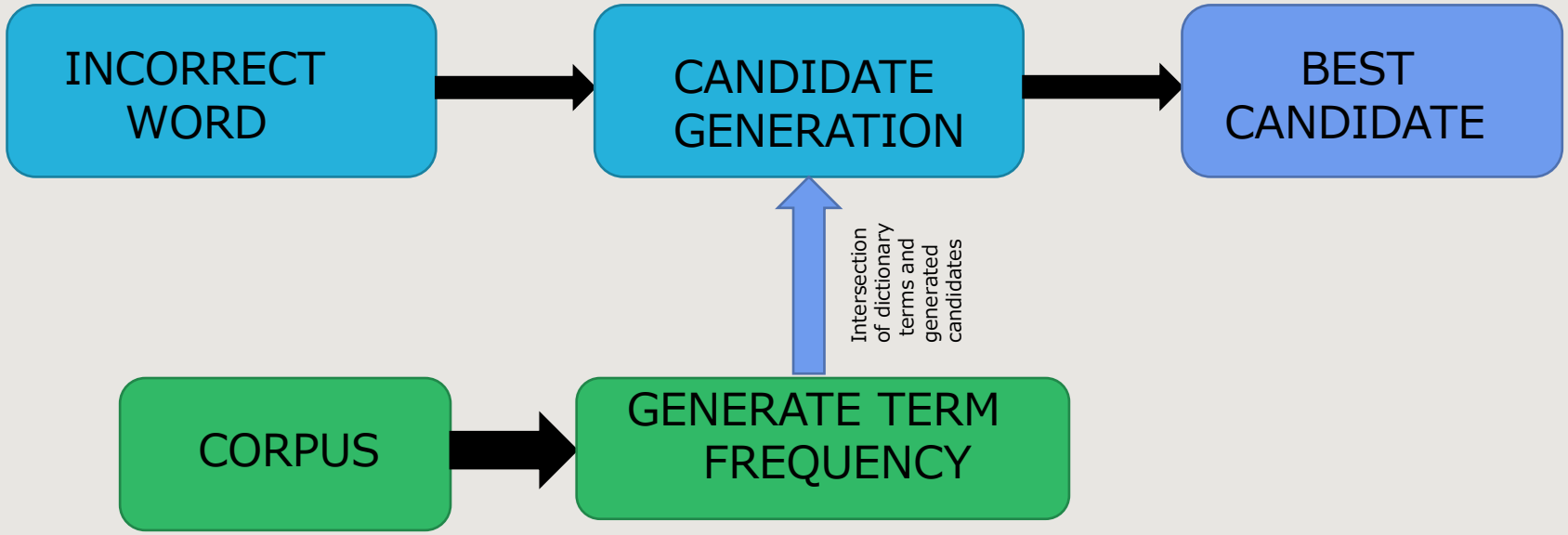
  Finds the best candidate based on term frequency.

- Unigram Noisy Channel Model –

  Uses Bayes theorem to filter the best candidate.

Language Model :-

Workflow –

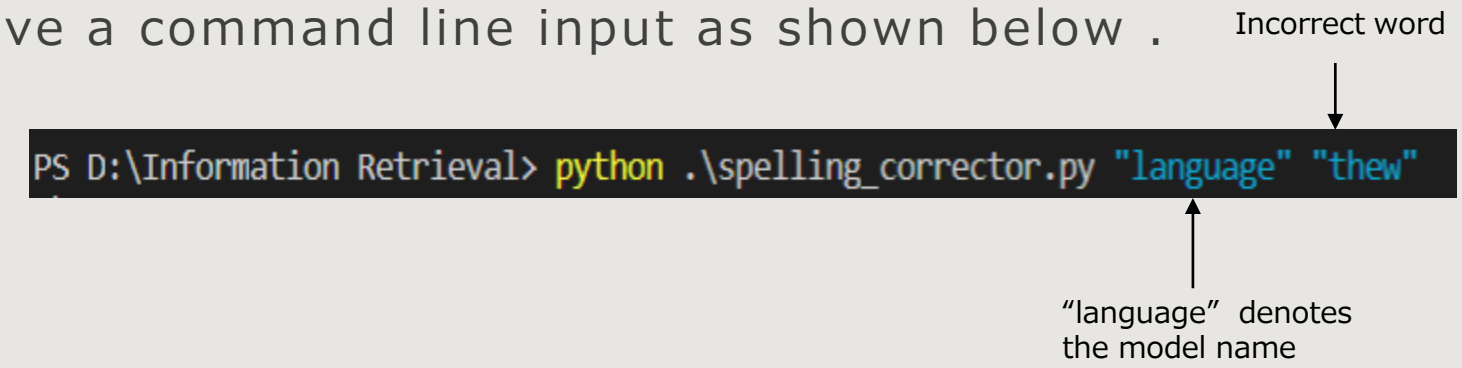


Language Model :-

- Takes incorrect word as an input.
- Generates all possible candidates within one and two edit distance.
- Since there can be many possible candidates, we consider only those candidates that are present in the corpus.
- Filters the best candidate that have maximum term frequency.

How it works :-

- Give a command line input as shown below .



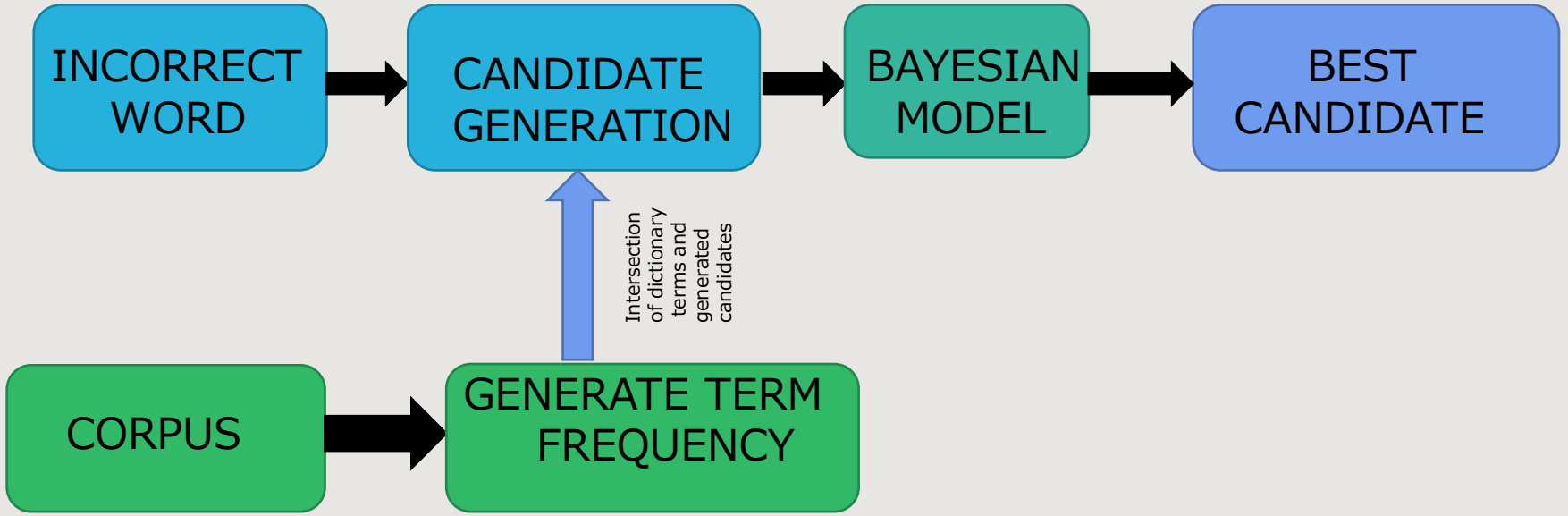
- It gives the best candidate as output.

```
PS D:\Information Retrieval> python .\spelling_corrector.py "language" "thew"
the
```

Here in above snippet, we can see that the correct word for misspelled word “thew” is “the”.

Noisy Channel Model :-

Workflow –



Noisy Channel Model

- Takes incorrect word as an input.
- Generates possible candidates within one edit distance.
- Since there can be many possible candidates so, It only considers those candidates that are there in the corpus.
- For each generated candidate, it calculates the likelihood probability based on the number of possible edits between incorrect word and generated candidate.

Error probability -

- P(*c* =*correct* | *w*=*incorrect*) = P(*w* | *c*)*P(*c*)

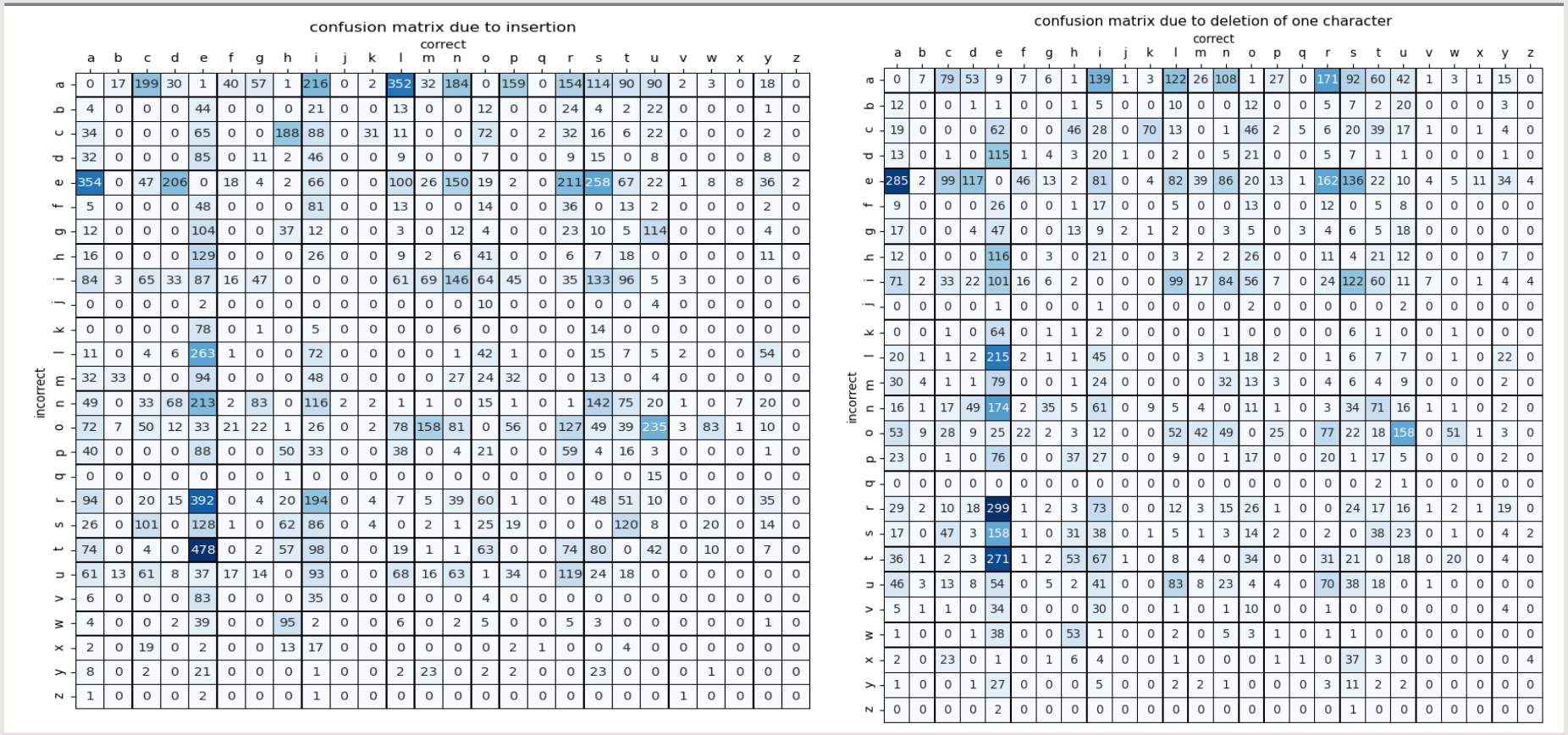
Where, P(*w* | *c*) is likelihood probability and P(*c*) is prior probability of each candidate.

- P(*w* | *c*) =

$$\begin{cases} \dfrac{del[c_{p-1}, c_p]}{chars[c_{p-1}, c_p]}, & \text{if deletion} \\ \dfrac{add[c_{p-1}, t_p]}{chars[c_{p-1}]}, & \text{if insertion} \\ \dfrac{sub[t_p, c_p]}{chars[c_p]}, & \text{if substitution} \\ \dfrac{rev[c_p, c_{p+1}]}{chars[c_p, c_{p+1}]}, & \text{if reversal} \end{cases}$$

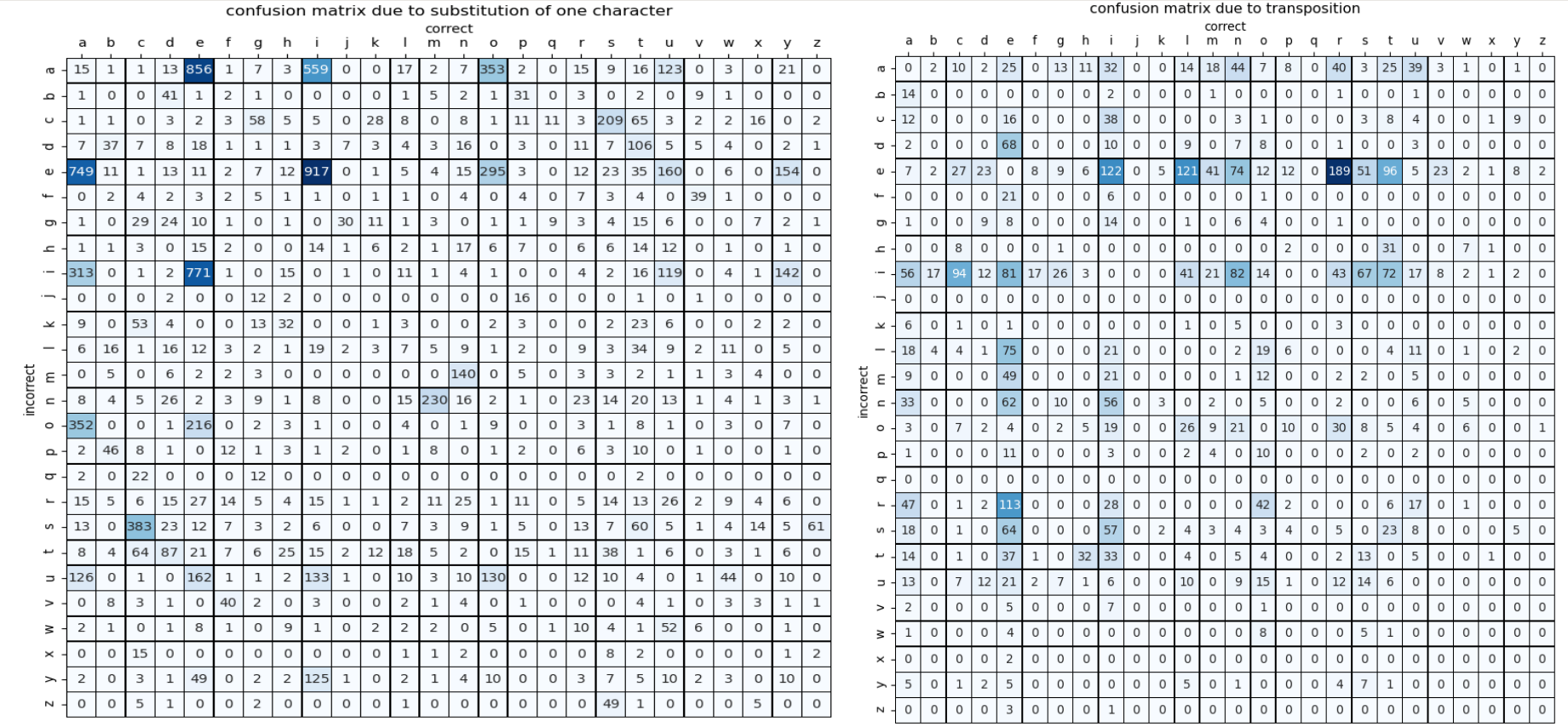
It calculates the above probability based on the confusion matrix as shown in next slide.

Confusion Matrix for Insertion and Deletion -



Chandan Kumar Roy | S20180010036

Confusion Matrix for Substitution and Transposition -



Chandan Kumar Roy | S20180010036

Calculation of error probability, Confusion Matrix -

```
del[x,y]  :  count(xy typed as x)
ins[x,y]  :  count(x typed as xy)
sub[x,y]  :  count(y typed as x)
trans[x,y]  :  count(xy typed as yx)
```

Note:- Insertion and deletion has been performed on previous character.

Unigram prior probability

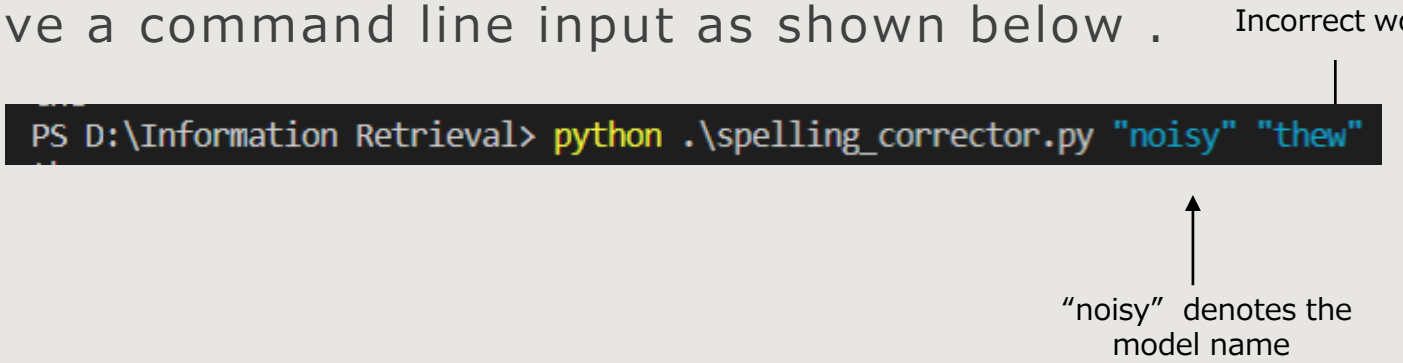
$P(c) = T(c)/N$

Where, N = Total number of terms in the dictionary,

$T(c)$ = Total count of candidate c.

How it works :-

- Give a command line input as shown below .



- It gives the best candidate as output.

```
PS D:\Information Retrieval> python .\spelling_corrector.py "noisy" "thew"
the
```

Here in above snippet, we can see that the correct word for misspelled word “thew” is “the”.

Testing -

- Accuracy for Language model is 35.3%.
- Accuracy for noisy channel model is 26.5%.

Language model outperforms noisy channel model because for noisy channel model, only candidates within one unit distance have been considered. While for language model, candidates within both one and two unit distance have been considered.

References -

- https://norvig.com/ngrams/
- https://www.2power3.com/rajendra/#slides
- https://norvig.com/ngrams/ch14.pdf
- https://web.stanford.edu/~jurafsky/slp3/B.pdf