- If someone wrongly types ‘ehat’ for ‘that’ or ‘soothie’ for ‘smoothie’, the Levenshtein-distance spelling corrector works well.
- Since I used the English word list/frequency data to build all the potential correct words, if the real correct word is not in the data, the Levenshtein-distance spelling corrector works poorly. In this case, we can add some words to the data.
- $$\begin{aligned}\hat{w} &= \arg\max_{w \in V} \log P(x|w) + \log P(w) \\ &= \arg\max_{w \in V} \hat{E} \log p + \log P(w)\end{aligned}$$

  Given this algorithm, the P(w) should be a priori probability of the author having written w. However, I used the frequency of a word in English (the number of a specific word divided by the total number of all words) as P(w). Therefore, if an author’s writing habit is not the same as most people (which means he doesn’t choose words that most people choose when he is writing), or he is writing for a specific academic field (for example, using medical terminology), the Levenshtein-distance spelling corrector works poorly. Since with the same edit distance, the word that will be returned depends on *logP(w),* (‘tge’ would have many words with 1 edit distance, the one with the highest prior probability wins), we should find out the true priori probability of the author having written w.
- For example, if the author typed “ehat is the weather like today?”, with my spelling corrector, the phrase will be corrected as “that is the weather like today?”, which is wrong. In this case, we also need to pay attention to context instead of just words.
- I chose p to be 0.00001. p is the weighting factor that can be tuned between the minimum distance and the prior probability. Therefore, there may a more suitable p for the spelling corrector.