Linguistic Project Abstract by Zelong Li (37718178) and Ricardo Liu (71222699)

Link to repository: https://github.com/XDDz123/TypePred

Our project is a typing prediction program that combines trigram and bigram probabilistic models to suggest words based on user input. This program contains two components written in python—model training and graphical user interface (GUI). The model training component receives a raw text file of well-formed sentences for training data. Sentences are first split into words, then special characters/symbols are removed with regex. In a sequential manner, the occurrence of each word is mapped to their immediate predecessor or two predecessors in python dictionaries to construct the bigram and trigram models, respectively. Both models are then trimmed to reduce size, where all entries except for the three highest occurrences are pruned from the model. This also serves to reduce noise in the model, since we are not processing the data on a sentence-by-sentence basis. Trimmed models are saved as JSON files, which would enable features such as using pre-trained models. The GUI component contains two primary sections. Users are expected to type in the text input field. As words are typed in, the program reads the user's most recently typed one or two words and accesses the bigram and trigram models loaded from the aforementioned JSON files once the user presses the spacebar. Special characters are trimmed from the user's typed words before accessing the respective models for the three most likely words (a.k.a. words with the highest occurrences) that follow. The suggested words are displayed as clickable labels on the bottom of the interface, where the user could click the respective label to enter the word into their text. Using sample training data from the internet of around 200MB, we believe the resulting model is sufficiently usable and within expectations. Notably, even after trimming, the trigram model was around 300MB in size. Running the bigram+trigram program took at least 2GB of system RAM. Undoubtedly, the space complexity of the project has plenty of room for improvement. On the other hand, the bigram model performed far more efficiently, where the model was only around 20MB in size and the bigram-only GUI took around 200MB of system RAM. Overall, bigram-only is a much more practical choice compared to bigram+trigram for our program.