# Proofreading Rewriting Midterm Pesentation

Team Dominatrix

Akkapaka Saikiran     Rwitaban Goswami     Shalabh Gupta

November 7, 2019

## spellcheck.py

### Algorithms Used

- Found all words which can be obtained from a given word by one of four operations : insertion, deletion, substitution, or transposition
- Sorted the suggestions based on two factors : Damerau Levenshtein Distance (using edit distance and keyboard distance) and word frequency from a corpus
- We used the idea of a spellcheck created by Peter Norvig (variation) and his BIG.txt for frequency check

### Why we did what we did

- The idea of Levesnshtien distance seemed a bit weak so we added in transposition and qwerty layout to estimate better corrections
- Even then suggestions were not up to the mark so we incorporated word frequency as well

## synonym_finder.py

### Algorithms Used

- Used an API called datamuse to find synonyms of certain words. Excluded articles, conjunctions, prepositions etc using nltk.pos_tag.
- Used an API called phrasefinder to find frequency of trigrams to decide whether it was a good synonym
- Used nltk.tokenise to split input into 'tokens'

### Why we did what we did

- We refused to use synonym finder of NLTK and instead opted to use an API because of the extra feature of left and right contexts which NLTK didn't offer. Also NLTK's list of synonyms was huge and we found it inefficient to work with
- To set preference order among synonyms we chose to sort them based on phrase (the trigram) frequency (More frequency $\rightarrow$ Better Suggestion).

## What the future holds in store

### Future Timeline

- By the end of endsems : Finish grammar check and read up Django.
- Midway through : Implement contextual spellcheck and punctuation.
- After endsems : Put the project together on the web using Django and implement voice changer

### Additional Features

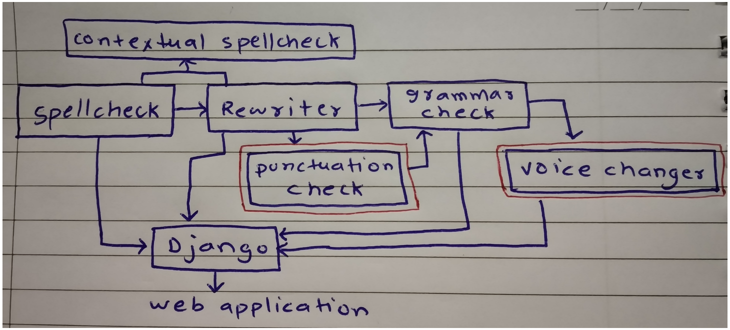
- Punctuation Check : Because we feel this matters more than grammar
- Voice Changer : To get interesting variations in sentences without changing their meaning. Also inverting parse trees sounds fun.

### Detailed Plan

- For grammar check we will use API for grammar extraction, then we will recursively parse our sentence. If it parses then done otherwise correct its grammar. Remaining Django based work will be done after learning Django through tutorials.

## Workflow + Thank You

This is the workflow we have in mind for our project. As of now the spellcheck and Rewriter tools have been constructed. Red boxes are our extra features.



# Thank You