Design Document for Text Conversion

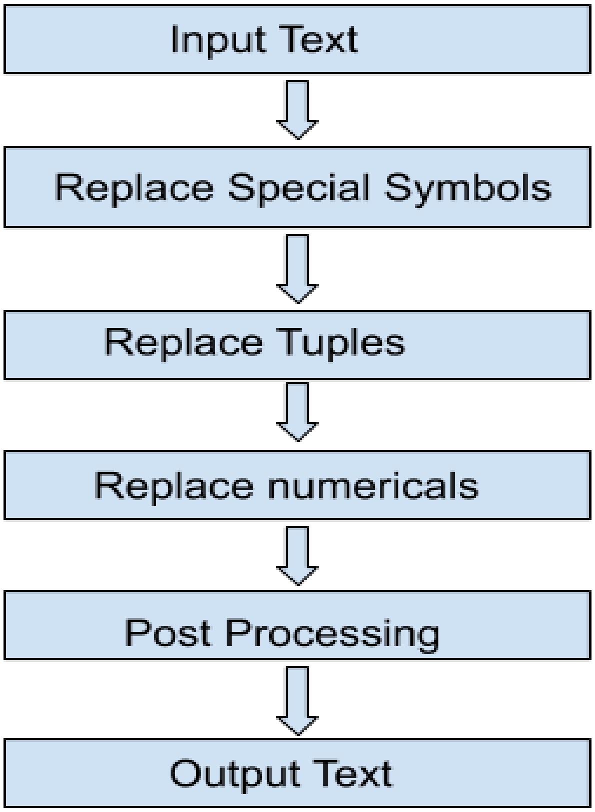
Feature Description:

- The functions implemented in the code in such away that each function attains a specific goal.
- For each conversion rule we created some feature map such as dictionaries and txt files, each function is responsible for a specific rule and it can access the corresponding mapping file and replaces the words.
- All the functions can be executed in some order for the convenience of input format i.e. output of a specific function goes as input to other function.
- No internal calling between functions.

Scalability: The code works even for inputs with large sizes. We can divide the input text data of larger sizes into smaller chunks and can be executed in parallel. Each paragraph or chunk can be treated as independent input. We can combine the output.

Adding of new conversion rules: As long as we can discover the new conversion rules we can add them by writing a function for that rule and create a mapping file like dictionaries or text files (if needed) without affecting the previous functionality. Call the function in proper sequence as per required input format.

Functions implemented and flow of execution:



read_file( ): reads the input text file and returns as a string to process further.

remove_whitespace( ): remove white spaces as part of pre processing.

get_special_symbol_dictionary( ): returns the dictionary used for replacing the special symbols

replace_special_symbols( ): replaces the word with the appropriate special symbol

get_numerical_terms_list( ): returns the list for identifying the words as numerals

get_numerical_type_dictionary( ): returns the dict for differentiate the types of numericals. i.e., digits(e.g. nine), non digit but numerical(e.g. fifty)

get_type( ): returns the type of the word returns 'D' if digit, returns 'N_ND' if it is non digit but numerical, returns 'others' if i not digit and not numerical

replace_numerical_terms1( ): replaces the numerical terms by its value

defintion_preprocessing( ): this conversion rule not implemented fully

get_tuple_dictionary( ): this function holds the tuple name as a key and its value as value (e.g. double = 2), returns the size of the tuple

replace_the_tuples1( ): replaces the words/numbers with multiples of the value corresponding to the tuple

complete_postprocessing( ): removes the gap between numbers, and do other post processing

dollar_processing( ): puts the currency in front of the figure by swaping the symbol and amount