

# 21-805-0704: Computational Linguistics Lab

## Lab Cycle 1

Date: 14/08/2024

1. Implement a simple rule-based Text tokenizer for the English language using regular expressions. Your tokenizer should consider punctuations and special symbols as separate tokens. Contractions like "isn't" should be regarded as 2 tokens - "is" and "n't". Also identify abbreviations (eg, U.S.A) and internal hyphenation (eg. ice-cream), as single tokens.
2. Design and implement a Finite State Automata(FSA) that accepts English plural nouns ending with the character 'y', e.g. boys, toys, ponies, skies, and puppies but not boies or toies or ponys. (Hint: Words that end with a vowel followed by 'y' are appended with 's' and will not be replaced with "ies" in their plural form).
3. Design and implement a Finite State Transducer(FST) that accepts lexical forms of English words(e.g. shown below) and generates its corresponding plurals, based on the e-insertion spelling rule  $e \Rightarrow e / \{x,s,z\}^{\wedge} \_ s^{\#}$   
^ is the morpheme boundary and # - word boundary

Input	Output
fox <sup>^</sup> s <sup>#</sup>	foxes
boy <sup>^</sup> s <sup>#</sup>	boys

4. Implement the *Minimum Edit Distance* algorithm to find the edit distance between any two given strings. Also, list the edit operations.

## Lab Cycle 2

Date: 21/08/24

5. Design and implement a statistical spell checker for detecting and correcting non-word spelling errors in English, using the bigram language model. Your program should do the following:
  - a. Tokenize the corpus and create a vocabulary of unique words.
  - b. Create a bi-gram frequency table for all possible bigrams in the corpus.
  - c. Scan the given input text to identify the non-word spelling errors
  - d. Generate the candidate list using 1 edit distance from the misspelled words
  - e. Suggest the best candidate word by calculating the probability of the given sentence using the bigram LM.

6. Implement a text classifier for sentiment analysis using the Naive Bayes theorem. Use Add-k smoothing to handle zero probabilities. Compare the performance of your classifier for k values 0.25, 0.75, and 1.