NLP Assignment

Unit 1

- Bishal Saini

1. Outline the three basic operations for Edit Operation

->

- In Natural language processing (NLP), the Edit distance also known as Levenshtein distance is a widely used concept to measure how different two strings are.

- The difference is calculated by the minimum number of operations required to transform one String into another.

- The three basic operations for Edit operation are as follow

i] Insertion :- - Adding a new character at any position in the string
- This operation increases the length of the String by one
- Used when a word is missing a character compared to the correct word.

Example :-
- Transform "Cat" -> "Cart"
  - Insert the letter 'r' between 'a' and 't'
- Transform 'helo' -> "hello"
  - Insert the missing 'l' to correct the spelling

- This is very common in typo corrections, where users often skip a letter while typing.

ii] Deletion :- - Removing a character from the String
- This operation decreases the length of the String by one
- Used when an extra character is present in the word compared to the target word.

Example :-
- Transform "cart" → "cat"
  Delete the letter 'r'.
- Transform "Committeee" → "committee"
  Delete one extra 'e'.

- Deletion is also very Common in errors where users press a key multiple times or accidentally add an unwanted character.

iii] ~~Subsit~~ Substitution :- - Replacing one character in the String with another character.
- This keeps the string length the same but modifies the content.
- Used when a letter is typed incorrectly or when recognizing noisy speech input.

Example :-
- Transform "bat" → "cat"
  - Substitute 'b' with 'c'
- Transform "recieve" → "receive"
  - Substitute 'i' with 'e'

- Substitution is frequent in spelling errors due to phonetic similarity.

2. Explain the Role of Noisy Channel Model and how does it Contribute

→

- In NLP, the noisy channel model is a powerful framework used to deal with uncertainty in communication.

- It is based on the idea that whenever a message is transmitted through a channel, the message may got disorted by noise.

- The receiver's task is to reconstruct the most ~~prob~~ probable original message from the noisy input.

- This concept was first introduced by Claude Shannon in information theory and later adapted in NLP for tasks such as Speech recognition, Spelling correction, and machine translation.

~~How~~ The Working of Noisy Channel Model

- The model is mathematically expressed as:

$$\hat{W} = \underset{w \in V}{\operatorname{argmax}} P(w/x) = \underset{w \in V}{\operatorname{argmax}} P(x/w).P(w)$$

Where :-
- x = observed (noisy) input
- w = intended (original) word or sentence
- $P(w)$ = prior probability of the word (from a language model)
- ~~P(w/x)~~
- $P(x/w)$ = likelihood of the noisy observation given the intended world (error model)

Role of Noisy Channel Model

i] Error Correction :- - Helps recover the original word when a spelling error or typo occurs.

- Example :- If user type 'hte', the model checks which valid word has the highest probability.
  - $P(hte|the) . P(the)$
  - $P(hte|hate) . P(hate)$
- Since "the" is far more frequent, the system suggests "the" as correction.

ii] Speech Recognition :- - Spoken words often contain background noise or pronunciation variations.

- Example :- If system hears sound like "their/there/they're", the model decides based on context ~~probabilit~~ probability which one is correct.

iii] Machine Translation :- - The model is used to estimate which target language sentence is most probable given a source sentence

- Example :- English sentence "I eight dinner early" could mean "I ate dinner early" or "I eight dinner early." The noisy channel model helps select "I ate dinner early" as it has higher probability and makes grammatical sense in English usage.

Unit 2

1. To improve the hidden Markov model performance suggest an approach for re-estimating the parameters

→

- The Hidden Markov Model (HMM) is one of the most important statistical models in NLP.
- It is widely used in speech recognition, part-of-speech (POS) tagging, named entity recognition and machine translation.

- An HMM Consists of three Key probability distributions:
  - i] Initial Probability (π) - the probability of starting in a particular hidden state.
  - ii] Transition Probability (A) - the probability of moving from one hidden state to another.
  - iii] Emission Probability (B) - the probability of observing a certain word (or symbol) given the hidden state.

- To improve HMM performance, we need to re-estimate these parameters from data so that the model better represents the actual language usage.

Approach :- Baum-Welch Algorithm (Expectation-Maximization)

- The most effective method for re-estimating HMM parameters is the Baum-welch algorithm,

Page No.
Date

which is a special case of the Expectation - Maximization (EM) algorithm.

This algorithm works in two main steps and is applied iteratively:

1. Expectation Step (E-Step) :- - Use the Forward - Backward algorithm to compute the expected counts of transitions and emissions in the trainning data.
- This step does not directly observe the hidden states but estimate their ~~proab~~ probabilities.

2. Maximization Step (M-step) :- - Recalculate the probabilities (π, A, B) using the expected counts from the E-step.
- Update :-
    - Initial probability = normalized probability of starting in a state.
    - Transition probability = expected number of transitions from one state to another / Expected total transitions from the state.
    - Emission probability = expected number of times a state emits a symbol / expected total emissions from the state.

- By repeating E-step and M-step, the algorithm increases the likelihood of the observed data, gradually improving the HMM's performance

Page No.
Date

2. What are the limitations of using probabilistic Context free grammar for computing the probability of a sentence & Does this parsing affect accuracy.

→

- Probabilistic Context-Free Grammars (PCFGs) extend standard Context-Free Grammars (CFGs) by assigning a probability to each production rule, allowing them to calculate the probability of a sentence and choose the most likely ~~pas~~ parse tree.

- The two major limitations of PCFGs stem from overly strong, unrealistic independence assumptions:

i] Context-Free Independence :- A ~~PCFG~~ PCFG assumes that the probability of applying a specific production rule is independent of the surrounding context within the parse tree.
- The Assumption :- If the rule is $A \rightarrow B\ C$, the probability $P(A \rightarrow B\ C / A)$ is the same regardless of where A appears in the tree or what words are nearby.
- The Reality (The Limitation) :- In natural language, the expansion of a pharse is highly dependent on its function within the sentence.

ii] Lexical Independence (Lack of Sensitivity to Words) :- A PCFG assumes that the probability of a syntactic structure is independent of the specific words (lexical items) that make up the phrases.

- The Assumption:- The probability of a Verb phrase (VP) expanding to a verb and a prepositional phrase (VP → V PP) is constant, no matter which specific verb is used.
- The Reality (The Limitation):- Grammar is highly sensitive to the head words. For instance, certain verbs subcategorize for specific complements (e.g., "devour" requires a direct object, while "sleep" does not).
- A PCFG cannot distinguish between VPs headed by different verbs, leading to grammatically nonsensical or low-probability parses being assigned a high probability.

- Effect on Parsing Accuracy.

- Yes, the limitations of PCFGs severely impact parsing accuracy.

- The inability to leverage context and specific lexical items forces a PCFG to make decisions that often lead to structurally incorrect or highly ambiguous parse trees.
- The most visible result is poor performance on two key challenges:

1 Prepositional Phrase Attachment Ambiguity:-
- This is a classic failure point. Given a sentence like: "She saw the man with the telescope," a parser must decide if the PP ("with the telescope") attaches to the verb saw (modifying the action) or the noun man (modifying the noun)

- A PCFG cannot use the specific words (saw vs. man vs telescope) to make this decision, as it only sees the probabilites of the generic rules VP -> V NP PP and NP -> NP PP. As result, it often chooses the structurally more frequent option, ignoring the strong lexical preference of the head words, thus lowering accuracy.

2. Coordination Ambiguity :- - PCFG also struggle with correctly identifying the scope of conjunctions (like and or).
- In the phrase "old men and women", a PCFG cannot easily determine if the adjective old modifies only men or both men and women - a distinction that is often resolved by word meanings, which the PCFG ignores.
- To address these limitations, modern, high-accuracy parsers rely on models like lexicalized PCFGs or Dependency parsers, which explicitly condition rule probabilites on the head word of each phrase, thus moving beyond the "Context-free" assumption.

Page No.
Date

3. Justify the need of Comparing distributions in Semantics give the Significance of Context window in distributionational Semantics.

→

- In Natural Language Processing (NLP), understanding the meaning of words is a fundamental Challenge.
- One of the most powerful approaches to represent meaning computationally is Distributions Semantics.

- Comparing distributions allows NLP systems to determine how similar or different two words or phrases are in terms of meaning. Here's why it's important:

1. To Measure Semantic Similarity :-
- By comparing word distributions, we can find how closely two words are related in meaning.

E.g :- The words "Car" and "automobile" often appear in similar contexts like "drive", "road", "engine", etc.

2. To Capture Contextual Meaning :-
- Words may have multiple meanings depending on Context. Comparing distributions helps the system understand these variations.

E.g :- The word "bank" occurs with "money", "loan", "cash" (financial sense) or "river", "water", "fishing" (geographical sense).

Page No.
Date

3. For Applications in NLP :-
- Words embeddings like Word2Vec, GloVe, and FastText are built on the principle of comparing word distributions
- This Comparison helps in machine translation, text classification, question answering, and semantic search.

Significance of Context Window in Distributional Semantics

- The context window plays a crucial role in how word meaning is captured
- A context window refers to the number of word ~~Meaning is captured~~ around a target word that are considered when building its meaning representation.

i] Small Context Window :- - Considers only nearby words (e.g., 2-3 words to the left and right)
- Captures syntactic or functional similarity words that play a similar grammatical role.

ii] Large Context Window :- - Includes more words around the target.
- Captures semantic or topical similarity words related by meaning rather than grammar.

iii] Balancing Context Window Size :- - A small window focuses on syntax, while a large window focuses on semantics.

- Therefore, Choosing the right window Size depends on the task:

  - POS tagging or grammar analysis -> Smaller window

  - Topic modeling or semantic search -> larger window.

[illegible]
11/10/25