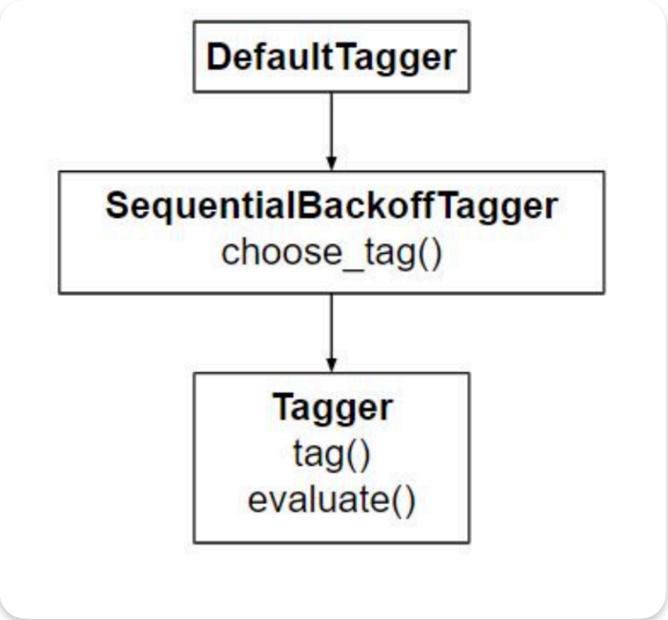
Def: Parts of Speech tagging is a linguistic activity in Natural Language Processing (NLP) wherein each word in a document is given a particular part of speech (adverb, adjective, verb, etc.) or grammatical category. Through the addition of a layer of syntactic and semantic information to the words, this procedure makes it easier to comprehend the sentence's structure and meaning.



Example

Sentence : The quick brown fox jumps over the lazy dog

- "The" is tagged as determiner (DT)
- "quick" is tagged as adjective (JJ)
- "brown" is tagged as adjective (JJ)
- "fox" is tagged as noun (NN)
- "jumps" is tagged as verb (VBZ)
- "over" is tagged as preposition (IN)
- "the" is tagged as determiner (DT)
- "lazy" is tagged as adjective (JJ)
- "dog" is tagged as noun (NN)

Workflow of processing POS Tagging

Tokenization :Divide the input text into discrete tokens, which are usually units of words or subwords. The first stage in NLP tasks is tokenization

Loading Language Models: To utilize a library such as NLTK or SpaCy, be sure to load the relevant language model. These models offer a foundation for comprehending a language's grammatical structure since they have been trained on a vast amount of linguistic data

Text Processing: If required, preprocess the text to handle special characters, convert it to lowercase, or eliminate superfluous information. Correct PoS labeling is aided by clear text.

Linguistic Analysis: To determine the text's grammatical structure, use linguistic analysis. This entails understanding each word's purpose inside the sentence, including whether it is an adjective, verb, noun, or other.

Part-of-Speech Tagging: To determine the text's grammatical structure, use linguistic analysis. This entails understanding each word's purpose inside the sentence, including whether it is an adjective, verb, noun, or other.

Results Analysis: Verify the accuracy and consistency of the PoS tagging findings with the source text. Determine and correct any possible problems or mistagging.

Code implementation using NLTK

```
# Importing the NLTK library
import nltk
from nltk.tokenize import word_tokenize
from nltk import pos_tag

# Sample text
text = "NLTK is a powerful library for natural language
processing."

# Performing PoS tagging
pos_tags = pos_tag(words)

# Displaying the PoS tagged result in separate lines
print("Original Text:")
print(text)

print("\nPoS Tagging Result:")
for word, pos_tag in pos_tags:
    print(f"{word}: {pos_tag}")
```

Output

```
Original Text:
NLTK is a powerful library for natural language
processing.
PoS Tagging Result:
NLTK: NNP
is: VBZ
a: DT
powerful: JJ
library: NN
for: IN
natural: JJ
language: NN
processing: NN
```

Code implementation using Space

```
!pip install spacy
!python -m spacy download en_core_web_sm
```

```
#importing libraries
import spacy

# Load the English language model
nlp = spacy.load("en_core_web_sm")

# Sample text
text = "SpaCy is a popular natural language processing
library."

# Process the text with SpaCy
doc = nlp(text)

# Display the PoS tagged result
print("Original Text: ", text)
print("PoS Tagging Result:")
for token in doc:
    print(f"{token.text}: {token.pos_}")
```

Output

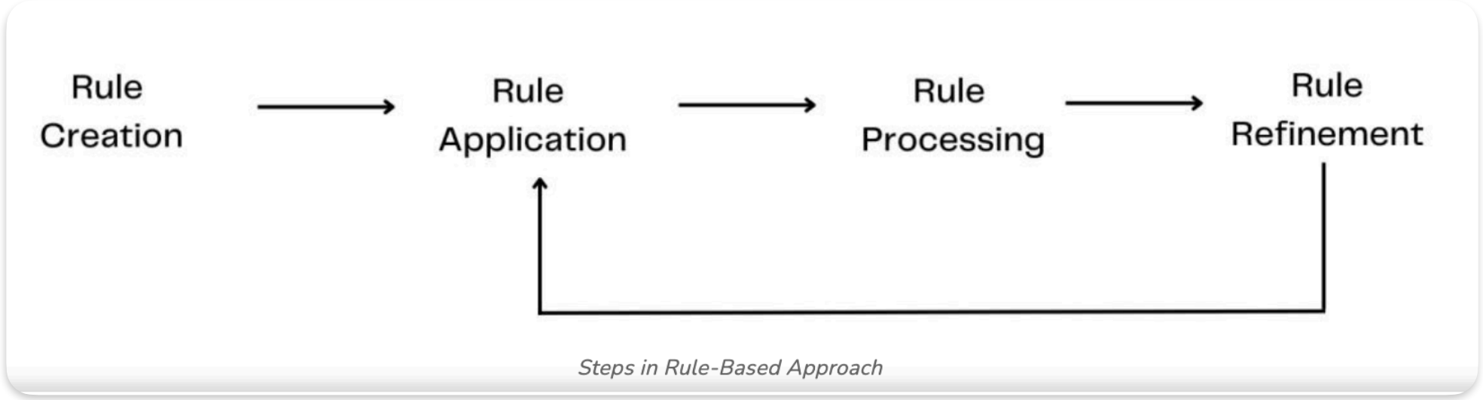
```
Original Text:  SpaCy is a popular natural language
processing library.
PoS Tagging Result:
SpaCy: PROPN
is: AUX
a: DET
popular: ADJ
natural: ADJ
language: NOUN
processing: NOUN
library: NOUN
.: PUNCT
```

Rule Based Approach in NLP

Based on linguistic rules and patterns

Rule-based approach is one of the oldest NLP methods in which predefined linguistic rules are used to analyze and process textual data. Rule-based approach involves applying a particular set of rules or patterns to capture specific structures, extract information, or perform tasks such as text classification and so on. Some common rule-based techniques include regular expressions and pattern matches

Rule Based Approach -workFlow



Steps in Rule-Based Approach

1. **Rule Creation:** Based on the desired tasks, domain-specific linguistic rules are created such as grammar rules, syntax patterns, semantic rules or regular expressions.
2. **Rule Application:** The predefined rules are applied to the inputted data to capture matched patterns.
3. **Rule Processing:** The text data is processed in accordance with the results of the matched rules to extract information, make decisions or other tasks.
4. **Rule refinement:** The created rules are iteratively refined by repetitive processing to improve accuracy and performance. Based on previous feedback, the rules are modified and updated when needed.