

**PROBLEM STATEMENT:**

Sentiment analysis, a pivotal task in Natural Language Processing (NLP), involves identifying and classifying sentiments expressed in text as positive, negative, or neutral. Traditional sentiment analysis methods often rely on complex machine learning algorithms and extensive annotated datasets, which can be resource-intensive and challenging to interpret. This project seeks to develop a more interpretable and efficient approach by leveraging Finite State Machines (FSMs) to perform sentiment analysis.

**PROJECT DESCRIPTION:**

This project aims to develop an efficient and interpretable method for sentiment analysis using Finite State Machines (FSMs). Sentiment analysis, or opinion mining, involves automatically identifying and classifying sentiments expressed in text data. By leveraging the structured and systematic nature of FSMs, this project seeks to provide a simple yet effective approach to recognize and classify sentiments as positive, negative, or neutral.

1. **FSM Design:** Construct a finite state machine tailored for sentiment analysis by defining states (neutral, positive, negative) and transitions based on sentiment-indicative words or phrases.
2. **Lexicon Development:** Compile comprehensive lexicons of words and phrases that signal positive and negative sentiments.
3. **Implementation:** Develop a Python-based FSM that processes text inputs and transitions through states to classify sentiments efficiently.

**Software Requirements:**

1. **Programming Language:**

Python is chosen due to its simplicity and extensive libraries for NLP tasks.

2. **Development Environment:**

An Integrated Development Environment (IDE) such as PyCharm, VS Code, or Jupyter Notebook for writing and testing the code.

3. **Libraries and Frameworks:**

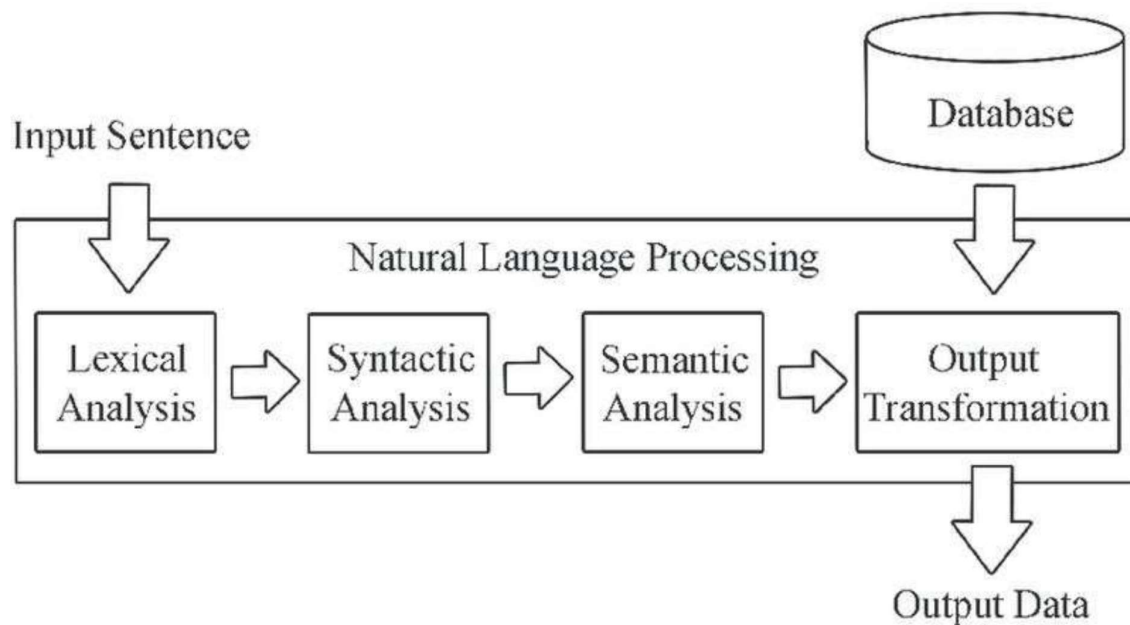
NLTK (Natural Language Toolkit): For text processing and sentiment lexicon development.

SciPy: For efficient text processing (optional but beneficial for advanced text parsing).

Pandas: For handling and manipulating text data.

NumPy: For numerical operations, if needed.

**FlowChart:**



**Code:**

```
class SentimentFSM:
    def __init__(self):
        self.states = ['Neutral', 'Positive', 'Negative', 'Final']
        self.current_state = 'Neutral'

        self.positive_lexicon = {'good', 'happy', 'excellent', 'great', 'fantastic', 'wonderful'}
        self.negative_lexicon = {'bad', 'sad', 'terrible', 'poor', 'awful', 'horrible'}

    def transition(self, word):
        if word in self.positive_lexicon:
            self.current_state = 'Positive'
        elif word in self.negative_lexicon:
            self.current_state = 'Negative'

    def process_sentence(self, sentence):
        self.current_state = 'Neutral' # Reset to initial state for each sentence
        words = sentence.split()

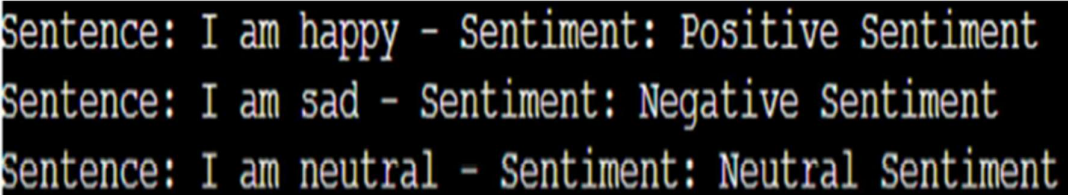
        for word in words:
            self.transition(word)

        self.current_state = 'Final'
```

```
return self.determine_sentiment()

def determine_sentiment(self):
    if self.current_state == 'Positive':
        return 'Positive Sentiment '
    elif self.current_state == 'Negative':
        return 'Negative Sentiment '
    else:
        return 'Neutral Sentiment '
```

**Output:**



```
Sentence: I am happy - Sentiment: Positive Sentiment
Sentence: I am sad - Sentiment: Negative Sentiment
Sentence: I am neutral - Sentiment: Neutral Sentiment
```