Vidyavardhini's College of Engineering and Technology

AY: 2025-26

Department of Artificial Intelligence & Data Science

Class:	**BE-AI&DS**	**Semester:**	**VII**
Course Code:	**CSDOL7011**	**Course Name:**	**NLP Lab**

Name of Student:	BARI ANKIT VINOD
Roll No. :	61
Experiment No.:	**10**
Title of the Experiment:	**Design and Development of a Real-World NLP Application**
Date of Performance:	
Date of Submission:	

CSDOL7011: Natural Language Processing Lab

Evaluation

Performance Indicator	Max. Marks	Marks Obtained
Performance	5	
Understanding	5	
Journal work and timely submission	10	
Total	20	

Performance Indicator	Exceed Expectations (EE)	Meet Expectations (ME)	Below Expectations (BE)
Performance	4-5	2-3	1
Understanding	4-5	2-3	1
Journal work and timely submission	8-10	5-8	1-4

Checked by

Name of Faculty :

Signature :

Date :

Aim: To design and implement a functional NLP application using appropriate models and techniques for solving a real-world problem such as sentiment analysis, text summarization, machine translation, information extraction, or question answering.

Objective: To design and implement a real-world NLP application using suitable models, techniques, and datasets.

Tools Required:

1. Python (Jupyter Notebook or Google Colab)
2. Relevant libraries based on the selected application:
    a. transformers, nltk, spacy, sklearn, textblob, flair, etc.

3. Dataset relevant to the chosen application (Kaggle, Hugging Face Datasets, UCI ML Repository)

Procedure:

1. Select an NLP Application:Choose a real-world use case (e.g., sentiment analysis for customer reviews, text summarization of news articles, etc.)
2. Define the Problem Statement and Objective: Clearly describe the scope, goal, and expected output of your application.
3. Collect and Preprocess Data:
    a. Source relevant data.
    b. Clean and preprocess the text (tokenization, stopword removal, etc.)
4. Select and Apply NLP Technique(s): Depending on the task, use appropriate methods:
        i. Rule-based
        ii. ML-based (Naive Bayes, SVM, etc.)
        iii. DL-based (LSTM, BERT, etc.)
        iv. Pretrained transformer models
5. Implement the Model and Application:
    a. Train or fine-tune the model (if needed).
    b. Create a simple interface (CLI or notebook cells) to interact with the system.

6. Evaluate the Application:
    a. Use suitable metrics (accuracy, ROUGE, BLEU, F1-score, etc.) depending on the task.
    b. Compare model performance or assess real-time usability.
7. Document the Design Choices:
    a. Model selection
    b. Preprocessing pipeline
    c. Evaluation approach
    d. Limitations and future scope

Description of the Experiment:

In this design-level project, students identify a real-world NLP problem and build a custom solution using appropriate data, techniques, and tools. This encourages creative thinking, technical synthesis, and project-based learning. Students apply their cumulative understanding of NLP to solve practical problems.

Detailed Description of the NLP Technique:

Examples of Applications and Techniques:

Application	**NLP Techniques**
Sentiment Analysis	Text classification, embeddings, transformers
Text Summarization	Extractive and abstractive models (BERT, T5)
Machine Translation	Sequence-to-sequence models (RNN, Transformer)
Information Extraction	Named Entity Recognition, pattern matching
Question Answering	Context-aware QA using BERT or RoBERTa

Students should research the best practices and models for the selected task and justify their design decisions.

Conclusion:

Architecture:
The AYNLP system is built using a modular architecture comprising three main components:

Preprocessing Module: Handles text cleaning, tokenization, and feature extraction using methods such as TF-IDF and embeddings.

Modeling Module: Implements deep learning-based NLP models (e.g., LSTM, Transformer-based embeddings) to perform tasks like text classification, sentiment analysis, or semantic similarity computation.

Interface Module: Provides a user-friendly interface for interaction, input, and displaying results in real-time.

This layered architecture ensures separation of concerns, scalability, and easy integration of new NLP models or datasets.

Results Obtained:
The system demonstrates accurate and efficient processing of textual data. Key results include:

High accuracy in text classification and semantic understanding tasks.

Effective handling of context and semantic relationships in text using embedding models.

Reliable performance across various NLP tasks, validating the robustness of the architecture.

Overall, the project illustrates that a well-structured architecture combined with advanced NLP techniques can deliver effective and interpretable results in practical applications.