Subject Code	Subject Name (Lab oriented Theory Courses)	Category	L	T	P	C
AI23632	NATURAL LANGUAGE PROCESSING	PC	3	0	2	4

Objectives:	
•	To introduce the fundamental concepts of Natural Language Processing (NLP for analysing words based on statistical measures and CORPUS.
•	To understand the principles of morphological analysis and language modeling using finite state machines and n-gram models.
•	To explore vector semantics and learn how to represent words and their relationships through embeddings and similarity measures.
•	To analyze and implement Hidden Markov Models (HMMs) and their applications in Part-Of-Speech (POS) tagging
•	To study the architecture of transformers and large language models, including pre-training and evaluation techniques.

UNIT-I	INTRODUCTION TO NATURAL LANGUAGE PROCESSING	9
Introduction to NLP - Various stages of NLP –NLP Pipeline, The Ambiguity of Language: Parts of Speech, Phrase Structure. Statistics Essential Information Theory: Entropy, perplexity, The relation to language: Cross entropy, Text Prepossessing: Character Encoding, Word Segmentation, Sentence Segmentation, Introduction to Corpora, Corpora Analysis		
UNIT-II	**MORPHOLOGY AND LANGUAGE MODELLING**	**9**
Inflectional and Derivation Morphology, Morphological analysis and generation using Finite State Automata and Finite State transducer.Bag of words, skip-gram, Continuous Bag-Of-Words, N gram model, n -gram Models over Sparse Data: Bins: Forming Equivalence Classes- - Statistical Estimators- Combining Estimators		
UNIT-III	**VECTOR SEMANTICS AND EMBEDDINGS**	**9**
Lexical Semantics-Vector Semantics-Words and Vectors-Cosine for measuring similarity- TF-IDF: Weighing terms in the vector- Pointwise Mutual Information (PMI) -Applications of the TF-IDF or PPMI vector models- Word2vec -Visualizing Embeddings - Semantic properties of embeddings - Bias and Embeddings - Evaluating Vector Models - Retrieval-Augmented Generation (RAG)		
UNIT-IV	**MARKOV MODEL AND POS TAGGING**	**9**
Markov Model: Hidden Markov model, Three Fundamental questions of HMM, Implementation properties, and Variants of HMMs, Multiple input observation. **POS**: The Information Sources in Tagging: Markov model taggers, Viterbi algorithm, Applying HMMs to POS tagging, Applications of Tagging.		
UNIT-V	**TRANSFORMERS AND LARGE LANGUAGE MODELS**	**9**
The Transformer - Attention-Transformer Blocks- Parallelizing computation using a single matrix X , The input: embeddings for token and position-The Language Modeling Head - Large Language Models : Large Language Models with Transformers -Sampling for LLM Generation - Pretraining Large Language Models -Evaluating Large Language Models		
Contact Hours	**:**	**45**

List of Experiments	
1.	Develop a morphological analyzer to process and analyze various sentence structures, including interrogative, declarative, and complex sentences with conjunctions. Perform word segmentation and sentence segmentation as part of the analysis. **Suggested Dataset/Corpus: Universal Dependencies (UD) English Treebank**
2.	Design a basic NLP pipeline to preprocess raw text data by performing tokenization, sentence segmentation, and part-of-speech (POS) tagging. Automate the pipeline to process large-scale text efficiently. **Suggested Dataset/Corpus: Universal Dependencies (UD) English Treebank**
3.	Implement a Named Entity Recognition (NER) system using Python libraries such as spaCy or NLTK. Utilize a pre-trained model to extract named entities, including people, organizations, and locations, from a text corpus. **Suggested Dataset/Corpus**: **CoNLL-2003 NER Dataset**

4.	Construct unigram, bigram, and trigram models to analyze their performance on sparse data. Compare the language models based on perplexity and their effectiveness in predicting word sequences. **Suggested Dataset/Corpus**: **The Brown Corpus**
5.	Implement n-gram language models (unigram, bigram, trigram, etc.) and apply smoothing techniques like Laplace smoothing to address data sparsity. Evaluate the models on a large text corpus for accuracy and perplexity. **Suggested Dataset/Corpus**: **Google Ngram Dataset**
6.	Design a spelling correction model using a combination of morphological rules and n-gram probabilities. Test the model on a dataset containing deliberately misspelled words and compare it to established spell-check systems. **Suggested Dataset/Corpus**: **Birkbeck Spelling Error Corpus**
7.	Implement the Term Frequency-Inverse Document Frequency (TF-IDF) model and use cosine similarity to compare the similarity between documents in a given corpus. Visualize the similarity matrix for better insight. **Suggested Dataset/Corpus**: **20 Newsgroups Dataset**
8.	Train a Word2Vec model on a given text corpus and visualize the resulting word embeddings using dimensionality reduction techniques like t-SNE or PCA. Analyze the semantic relationships between words in the embeddings. **Suggested Dataset/Corpus**: **Text8 Dataset**
9.	Build a Hidden Markov Model (HMM) for part-of-speech (POS) tagging. Train the model on a tagged corpus and evaluate its accuracy on a test dataset. **Suggested Dataset/Corpus**: **Universal Dependencies (UD) Treebank**
10.	Use a pre-trained Transformer model (e.g., BERT) to build a sentiment analysis model. Fine-tune the model on a dataset of tweets, classify sentiment (positive, neutral, negative), and evaluate its performance using accuracy and F1-score. **Suggested Dataset/Corpus**: **Sentiment140 Dataset**
11	Use a pre-trained language model to perform sentiment analysis or keyword extraction on a dataset of WhatsApp chat and E mail data. Analyze the conversational patterns, emotions, and key topics discussed in the chats. (Multiple languages such as English, Tamil, etc.) **Suggested Dataset/Corpus**: **WhatsApp Chat Export (User-Generated Data)**
12	Use a pre-trained language model to perform sentiment analysis or keyword extraction from E mail data. Analyze the conversational patterns, information, and key topics from Email Message. (Multiple languages such as English, Tamil, etc.) **Suggested Dataset/Corpus**: **Email Data (User-Generated Data)**
13	Implement a question-answering system using a pre-trained BERT model. Input a passage and a question, and use the model to extract the correct answer from the passage. Evaluate the system on accuracy and relevance of the answers. **Suggested Dataset/Corpus**: **SQuAD (Stanford Question Answering Dataset)**
14	**Mini Project** • Choose a Topic: Identify a deep learning problem of interest, such as image classification, text generation, or anomaly detection. • Research related works using platforms like Google Scholar. • Dataset Selection: Find or collect a suitable dataset from sources like Kaggle or UCI. Ensure it is relevant, well-sized, and consider preprocessing requirements. • Develop Methodology: Start with baseline models, then experiment with advanced architectures (e.g., CNNs, Transformers). Use frameworks like TensorFlow or PyTorch. • Implementation & Evaluation: Train models and evaluate performance using appropriate metrics (e.g., accuracy, F1-score). Document findings systematically. • Discuss & Present: Analyze results, highlight challenges, and present your work with clear insights and