# Natural Language Processing

Target Audience: Undergraduate students

## Module Outline

# Natural Language Processing (NLP) - 8-Week Undergraduate Curriculum Outline

This curriculum provides an 8-week introduction to Natural Language Processing for undergraduate students. It covers foundational concepts, traditional machine learning approaches, and an introduction to modern deep learning techniques, culminating in discussions on applications and ethical considerations.

---

### Module 1: Foundations of NLP & Text Processing

#### Week 1: Introduction to NLP & Text Preprocessing

* \*\*Title:\*Unlocking Language: Introduction to NLP & Text Preprocessing
* \*\*Topics:\*\*
* What is Natural Language Processing? History, challenges, and real-world applications (e.g., search engines, spam detection, virtual assistants).
* Basic components of NLP systems.
* Text Representation: Raw text, character encoding.
* Text Preprocessing Techniques:
* Tokenization (word, sentence).
* Normalization (case folding, punctuation removal).
* Stemming (Porter, Lancaster).
* Lemmatization (WordNet Lemmatizer).
* Stop words removal.
* Regular Expressions for text pattern matching.
* Introduction to Python libraries for NLP (e.g., NLTK, spaCy basics).
* \*\*Learning Outcomes:\*\*
* Understand the fundamental concepts and broad applications of Natural Language Processing.
* Identify and apply various text preprocessing techniques to raw text data.
* Utilize regular expressions for basic text manipulation and pattern extraction.
* Set up a Python environment for NLP tasks and perform basic operations using relevant libraries.

#### Week 2: Language Modeling & Text Classification Basics

* \*\*Title:\*Understanding Language Patterns: N-grams & Basic Text Classification
* \*\*Topics:\*\*
* Introduction to Language Modeling: Purpose and applications (e.g., spelling correction, predictive text).
* N-gram Language Models: Unigrams, Bigrams, Trigrams.
* Calculating probabilities: Maximum Likelihood Estimation.
* Smoothing techniques (Laplace, Add-k smoothing - conceptual).
* Perplexity as an evaluation metric for language models.
* Text Representation for Machine Learning:
* Bag-of-Words (BoW) model.
* Term Frequency-Inverse Document Frequency (TF-IDF).
* Introduction to Text Classification: Concepts and simple examples.
* \*\*Learning Outcomes:\*\*
* Explain the concept of N-gram language models and their role in NLP.
* Calculate N-gram probabilities and understand the concept of perplexity.
* Implement and apply Bag-of-Words and TF-IDF for text vectorization.
* Grasp the basic principles of text classification as an NLP task.

---

### Module 2: Traditional NLP Models & Machine Learning for Text

#### Week 3: Part-of-Speech Tagging & Introduction to Word Embeddings

* \*\*Title:\*Deconstructing Sentences: POS Tagging & Word Meanings
* \*\*Topics:\*\*
* Part-of-Speech (POS) Tagging: Importance and applications.
* Rule-based vs. Statistical POS Tagging (brief overview).
* Hidden Markov Models (HMMs) for sequence tagging (conceptual understanding of state transitions and emissions).
* Evaluation of POS taggers.
* Introduction to Distributional Semantics.
* Word Embeddings: The concept of dense vector representations.
* Word2Vec (Skip-gram and CBOW - intuition).
* GloVe (Global Vectors for Word Representation - intuition).
* Exploring pre-trained word embeddings.
* \*\*Learning Outcomes:\*\*
* Understand the purpose and challenges of Part-of-Speech tagging.
* Explain the intuition behind statistical sequence tagging models like HMMs.
* Grasp the concept of word embeddings and their advantage over sparse representations.
* Use pre-trained word embeddings for basic tasks.

#### Week 4: Syntactic Parsing & Dependency Parsing

* \*\*Title:\*Sentence Structure: Constituency & Dependency Parsing
* \*\*Topics:\*\*
* Introduction to Syntax and Grammars in NLP.
* Context-Free Grammars (CFGs): Rules, Parse Trees.
* Constituency Parsing: Top-down vs. Bottom-up parsing strategies (conceptual).
* Introduction to the CKY algorithm (conceptual overview for parsing CFGs).
* Dependency Parsing: Introduction to dependency relations (e.g., subject, object, modifier).
* Universal Dependencies (UD) framework (overview of common dependency types).
* Applications of syntactic and dependency parsing (e.g., information extraction, machine translation).
* \*\*Learning Outcomes:\*\*
* Understand the role of grammars in representing sentence structure.
* Differentiate between constituency and dependency parsing.
* Interpret parse trees generated by constituency and dependency parsers.
* Identify common dependency relations within sentences.

#### Week 5: Traditional Machine Learning for NLP Tasks

* \*\*Title:\*Building Classifiers: Machine Learning for NLP
* \*\*Topics:\*\*
* Review of Supervised Learning for Classification.
* Feature Engineering for Text: Beyond BoW/TF-IDF (e.g., N-gram features, length, special characters counts).
* Classification Algorithms:
* Naive Bayes Classifier (for text classification).
* Support Vector Machines (SVMs) for text data.
* Logistic Regression (brief overview).
* Model Evaluation Metrics:
* Accuracy, Precision, Recall, F1-score.
* Confusion Matrix.
* Cross-validation.
* Case Study: Sentiment Analysis or Spam Detection using traditional ML.
* \*\*Learning Outcomes:\*\*
* Apply traditional machine learning algorithms (Naive Bayes, SVMs) to solve NLP classification problems.
* Perform basic feature engineering for text data.
* Evaluate the performance of NLP models using standard metrics.
* Implement a simple text classification project from data preparation to evaluation.

---

### Module 3: Deep Learning for NLP & Applications

#### Week 6: Neural Networks for NLP - Basics

* \*\*Title:\*The Neural Shift: Introduction to Deep Learning for NLP
* \*\*Topics:\*\*
* Limitations of traditional ML for complex language patterns.
* Introduction to Neural Networks:
* Neurons, activations functions (ReLU, Sigmoid, Softmax).
* Perceptrons and Multi-Layer Perceptrons (MLPs).
* Feedforward Networks.
* Training Neural Networks: Loss functions, Optimization (Gradient Descent, Adam - conceptual).
* Backpropagation (conceptual understanding, not mathematical derivation).
* Introduction to deep learning frameworks (e.g., Keras/TensorFlow or PyTorch) for text tasks.
* Using word embeddings as input to neural networks.
* \*\*Learning Outcomes:\*\*
* Understand the fundamental concepts of neural networks and their relevance to NLP.
* Explain the basic architecture and training process of a feedforward neural network.
* Build and train a simple multi-layer perceptron for a text classification task using a deep learning framework.

#### Week 7: Recurrent Neural Networks (RNNs) and LSTMs

* \*\*Title:\*Sequential Understanding: Recurrent Neural Networks
* \*\*Topics:\*\*
* Challenges of processing sequential data with feedforward networks.
* Recurrent Neural Networks (RNNs): Architecture, hidden states, unrolling.
* Vanishing and Exploding Gradients problem in RNNs.
* Long Short-Term Memory (LSTM) Networks: Gate mechanisms (input, forget, output gates).
* Gated Recurrent Units (GRUs): Simplified version of LSTMs.
* Bidirectional RNNs/LSTMs.
* Sequence-to-Sequence (Seq2Seq) models (Encoder-Decoder architecture - conceptual).
* \*\*Learning Outcomes:\*\*
* Explain the need for recurrent neural networks for sequence modeling.
* Understand the architecture and operation of RNNs, LSTMs, and GRUs.
* Identify scenarios where LSTMs/GRUs are superior to simple RNNs.
* Build a basic RNN/LSTM model for sequence classification or generation (e.g., text generation, POS tagging).

#### Week 8: Attention Mechanisms & Transformers (Introduction) + Ethical NLP

* \*\*Title:\*The Modern NLP Landscape: Attention, Transformers & Ethics
* \*\*Topics:\*\*
* Limitations of Seq2Seq models without attention.
* Attention Mechanism: Intuition, how it improves sequence modeling.
* Introduction to Transformers: Self-attention mechanism, Multi-head attention (conceptual).
* Encoder-Decoder architecture of Transformers (high-level overview).
* Overview of Transformer-based models: BERT, GPT (high-level understanding of pre-training and fine-tuning).
* Key NLP Applications: Machine Translation, Text Summarization, Question Answering, Chatbots/Conversational AI.
* Ethical Considerations in NLP: Bias in data and models, fairness, privacy, explainability, misuse of NLP.
* \*\*Learning Outcomes:\*\*
* Grasp the conceptual intuition behind attention mechanisms and their benefits.
* Understand the high-level architecture of Transformer networks.
* Recognize the capabilities and applications of state-of-the-art NLP models like BERT and GPT.
* Discuss the ethical implications and potential biases in NLP systems.
* Identify current trends and future directions in Natural Language Processing.

## Learning Material

# Natural Language Processing (NLP) - 8-Week Undergraduate Curriculum Learning Materials

This document provides a detailed list of learning materials for each week of the Natural Language Processing curriculum, structured to support the outlined topics and learning outcomes.

---

### Module 1: Foundations of NLP & Text Processing

#### Week 1: Introduction to NLP & Text Preprocessing

* \*\*Lecture Notes:\*\*
* \*\*PDF/Slides:\*"Introduction to NLP: History, Challenges, and Applications."
* \*\*PDF/Slides:\*"Text Representation & Character Encoding: ASCII, Unicode, UTF-8."
* \*\*PDF/Slides:\*"Core Text Preprocessing Techniques: Tokenization, Normalization, Stemming, Lemmatization, Stop Words Removal."
* \*\*PDF/Slides:\*"Mastering Text Patterns: Regular Expressions for NLP."
* \*\*Reading:\*Chapter 1 from "Speech and Language Processing" (Jurafsky & Martin) - Introduction.
* \*\*Reading:\*NLTK Book, Chapter 3 (Processing Raw Text).
* \*\*Code Examples (Python Notebooks/Scripts):\*\*
* \*\*`01\_basic\_string\_ops.ipynb`\*\*: Demonstrates Python string methods (split, lower, replace).
* \*\*`02\_nltk\_spacy\_tokenization.ipynb`\*\*: Word and sentence tokenization examples using NLTK and spaCy.
* \*\*`03\_text\_normalization.ipynb`\*\*: Code for case folding, punctuation removal, and handling numbers.
* \*\*`04\_stemming\_lemmatization.ipynb`\*\*: Examples of Porter, Lancaster stemmers, and WordNet Lemmatizer.
* \*\*`05\_stopwords\_removal.ipynb`\*\*: Removing common stop words from text.
* \*\*`06\_regex\_for\_text.ipynb`\*\*: Practical regex patterns for email, phone numbers, dates, and custom patterns.
* \*\*`07\_full\_preprocessing\_pipeline.ipynb`\*\*: Integrates all preprocessing steps into a reusable function.
* \*\*Case Studies/Practical Exercises:\*\*
* \*\*Exercise:\*Preprocess a provided raw text document (e.g., an excerpt from Project Gutenberg) by applying all learned techniques.
* \*\*Exercise:\*Use regular expressions to extract specific entities (e.g., all dates, hashtags, URLs) from a dataset of tweets or forum posts.
* \*\*Mini-Project:\*Build a simple "word counter" that takes raw text, preprocesses it, and counts unique words (excluding stop words).

#### Week 2: Language Modeling & Text Classification Basics

* \*\*Lecture Notes:\*\*
* \*\*PDF/Slides:\*"Introduction to Language Models: N-grams and Probabilities."
* \*\*PDF/Slides:\*"Evaluating Language Models: Perplexity and Smoothing Techniques (Conceptual)."
* \*\*PDF/Slides:\*"Text Vectorization: Bag-of-Words and TF-IDF Explained."
* \*\*PDF/Slides:\*"Foundations of Text Classification."
* \*\*Reading:\*NLTK Book, Chapter 2 (Accessing Text Corpora and Lexical Resources).
* \*\*Reading:\*NLTK Book, Chapter 6 (Categorizing and Tagging Words).
* \*\*Code Examples (Python Notebooks/Scripts):\*\*
* \*\*`01\_ngram\_generation.ipynb`\*\*: Python code to generate unigrams, bigrams, and trigrams from a sentence or corpus.
* \*\*`02\_ngram\_probability\_calculation.ipynb`\*\*: Demonstrates Maximum Likelihood Estimation for N-gram probabilities.
* \*\*`03\_laplace\_smoothing\_concept.ipynb`\*\*: Simple code example illustrating Laplace smoothing.
* \*\*`04\_bow\_tfidf\_vectorization.ipynb`\*\*: Using `CountVectorizer` and `TfidfVectorizer` from scikit-learn for text representation.
* \*\*`05\_basic\_text\_classification\_prep.ipynb`\*\*: Preparing a small dataset for text classification using BoW/TF-IDF.
* \*\*Case Studies/Practical Exercises:\*\*
* \*\*Exercise:\*Implement a basic N-gram language model from scratch (without `nltk.lm`) to predict the next word in short sentences.
* \*\*Exercise:\*Compare the vector representations generated by BoW and TF-IDF for a small set of customer reviews, highlighting differences.
* \*\*Mini-Project:\*Create a simple text classification pipeline: load a small dataset (e.g., spam vs. ham SMS messages), vectorize using TF-IDF, and prepare data for a classifier (conceptual, actual classification in Week 5).

---

### Module 2: Traditional NLP Models & Machine Learning for Text

#### Week 3: Part-of-Speech Tagging & Introduction to Word Embeddings

* \*\*Lecture Notes:\*\*
* \*\*PDF/Slides:\*"Part-of-Speech Tagging: Methods and Applications."
* \*\*PDF/Slides:\*"Sequence Tagging: Intuition of Hidden Markov Models (HMMs)."
* \*\*PDF/Slides:\*"Beyond Sparse: Introduction to Distributional Semantics and Word Embeddings."
* \*\*PDF/Slides:\*"Word2Vec (Skip-gram, CBOW) and GloVe: The Core Ideas."
* \*\*Reading:\*NLTK Book, Chapter 5 (Categorizing and Tagging Words - POS Tagging).
* \*\*Video:\*"Word2Vec Explained: Negative Sampling" (Conceptual).
* \*\*Code Examples (Python Notebooks/Scripts):\*\*
* \*\*`01\_nltk\_spacy\_pos\_tagging.ipynb`\*\*: Demonstrates POS tagging with NLTK and spaCy, including visualization.
* \*\*`02\_exploring\_pos\_tagging\_eval.ipynb`\*\*: Basic evaluation of POS tagger performance (conceptual).
* \*\*`03\_loading\_pretrained\_word\_embeddings.ipynb`\*\*: Loading and exploring pre-trained GloVe or Word2Vec embeddings (e.g., via `gensim` or spaCy).
* \*\*`04\_word\_similarity\_analogy.ipynb`\*\*: Calculating cosine similarity between words, demonstrating analogies (e.g., "king - man + woman = queen").
* \*\*`05\_visualizing\_embeddings.ipynb`\*\*: Simple PCA/t-SNE visualization of a small set of word embeddings.
* \*\*Case Studies/Practical Exercises:\*\*
* \*\*Exercise:\*Analyze the POS tags of a paragraph from a novel and a paragraph from a scientific paper. Discuss differences in word usage patterns.
* \*\*Exercise:\*Use a pre-trained word embedding model to find the most similar words to a given list of terms (e.g., "doctor," "engineer," "artist").
* \*\*Mini-Project:\*Use POS tagging to identify all nouns, verbs, and adjectives in a given text and count their frequencies.

#### Week 4: Syntactic Parsing & Dependency Parsing

* \*\*Lecture Notes:\*\*
* \*\*PDF/Slides:\*"Introduction to Syntax and Context-Free Grammars (CFGs)."
* \*\*PDF/Slides:\*"Constituency Parsing: Top-down, Bottom-up, and the CKY Algorithm (Conceptual)."
* \*\*PDF/Slides:\*"Dependency Parsing: Relations and Universal Dependencies (UD)."
* \*\*PDF/Slides:\*"Applications of Syntactic and Dependency Parsing."
* \*\*Reading:\*NLTK Book, Chapter 8 (Analyzing Sentence Structure).
* \*\*Resource:\*Universal Dependencies Documentation (Explore common relations).
* \*\*Code Examples (Python Notebooks/Scripts):\*\*
* \*\*`01\_nltk\_cfg\_parser.ipynb`\*\*: Defining simple CFGs and using NLTK's `ChartParser` for constituency parsing.
* \*\*`02\_spacy\_dependency\_parsing.ipynb`\*\*: Demonstrating spaCy's dependency parser and visualizing parse trees.
* \*\*`03\_extracting\_subject\_object.ipynb`\*\*: Using dependency parsing to extract simple subject-verb-object triplets.
* \*\*Case Studies/Practical Exercises:\*\*
* \*\*Exercise:\*Given a simple CFG, manually draw the parse tree for a specific sentence.
* \*\*Exercise:\*Use spaCy to parse 5 different sentences and identify the head word and dependency relation for each token.
* \*\*Mini-Project:\*Develop a small script that uses dependency parsing to identify commands or actions (e.g., "turn on lights", "play music") from short user queries.

#### Week 5: Traditional Machine Learning for NLP Tasks

* \*\*Lecture Notes:\*\*
* \*\*PDF/Slides:\*"Supervised Learning Refresher: Classification Principles."
* \*\*PDF/Slides:\*"Feature Engineering for Text: Beyond Simple Vectorization."
* \*\*PDF/Slides:\*"Classic ML Algorithms for Text: Naive Bayes, SVMs, Logistic Regression."
* \*\*PDF/Slides:\*"Model Evaluation Metrics: Accuracy, Precision, Recall, F1-score, Confusion Matrix, Cross-validation."
* \*\*Reading:\*Scikit-learn documentation for `CountVectorizer`, `TfidfVectorizer`, `MultinomialNB`, `SVC`.
* \*\*Video:\*"Understanding the Confusion Matrix" (Conceptual).
* \*\*Code Examples (Python Notebooks/Scripts):\*\*
* \*\*`01\_advanced\_text\_features.ipynb`\*\*: Examples of creating features like text length, word count, number of uppercase words, specific N-grams.
* \*\*`02\_naive\_bayes\_text\_classification.ipynb`\*\*: Implementing Naive Bayes for a text classification task (e.g., spam detection) using scikit-learn.
* \*\*`03\_svm\_logistic\_regression\_text.ipynb`\*\*: Applying SVM and Logistic Regression to the same text classification task.
* \*\*`04\_evaluation\_metrics\_cross\_validation.ipynb`\*\*: Calculating and interpreting classification report, confusion matrix, and performing k-fold cross-validation.
* \*\*Case Studies/Practical Exercises:\*\*
* \*\*Comprehensive Project: Sentiment Analysis:\*\*
* \*\*Dataset:\*IMDb Movie Reviews or Twitter Sentiment dataset.
* \*\*Task:\*Classify movie reviews as positive or negative.
* \*\*Steps:\*\*

1. Load and preprocess data.

2. Vectorize text using TF-IDF.

3. Train and evaluate Naive Bayes, SVM, and Logistic Regression models.

4. Compare model performance using various metrics.

5. Analyze misclassifications using the confusion matrix.

* \*\*Exercise:\*Experiment with different feature engineering approaches (e.g., character N-grams vs. word N-grams) and observe their impact on model performance.

---

### Module 3: Deep Learning for NLP & Applications

#### Week 6: Neural Networks for NLP - Basics

* \*\*Lecture Notes:\*\*
* \*\*PDF/Slides:\*"Limitations of Traditional ML for Complex Language."
* \*\*PDF/Slides:\*"Neural Network Fundamentals: Neurons, Activation Functions, Perceptrons, MLPs."
* \*\*PDF/Slides:\*"Training Neural Networks: Loss Functions, Optimization (Gradient Descent, Adam)."
* \*\*PDF/Slides:\*"Backpropagation: The Learning Algorithm (Conceptual)."
* \*\*PDF/Slides:\*"Introduction to Deep Learning Frameworks: Keras/TensorFlow or PyTorch."
* \*\*PDF/Slides:\*"Integrating Word Embeddings into Neural Networks."
* \*\*Reading:\*Deep Learning Book (Goodfellow et al.), Chapter 6 (Deep Feedforward Networks - select sections).
* \*\*Code Examples (Python Notebooks/Scripts):\*\*
* \*\*`01\_simple\_perceptron\_conceptual.ipynb`\*\*: A very basic conceptual implementation of a single perceptron.
* \*\*`02\_keras\_mlp\_text\_classification.ipynb`\*\*: Building and training a simple Multi-Layer Perceptron (MLP) for text classification using Keras/TensorFlow.
* \*\*`03\_embedding\_layer\_input.ipynb`\*\*: Demonstrates how to use a pre-trained word embedding layer (e.g., GloVe) as input to an MLP.
* \*\*`04\_experimenting\_activations\_optimizers.ipynb`\*\*: Code to experiment with different activation functions (ReLU, Sigmoid, Softmax) and optimizers (Adam, SGD).
* \*\*Case Studies/Practical Exercises:\*\*
* \*\*Mini-Project: MLP for Sentiment Analysis:\*\*
* \*\*Dataset:\*Revisit the sentiment analysis dataset from Week 5.
* \*\*Task:\*Build an MLP model using Keras/TensorFlow with an embedding layer (either pre-trained or learned from scratch).
* \*\*Steps:\*\*

1. Preprocess and tokenize text.

2. Create an integer sequence representation for sentences.

3. Define, compile, and train the MLP model.

4. Evaluate its performance and compare with traditional ML models.

* \*\*Exercise:\*Experiment with increasing the number of layers or neurons in the MLP and observe the effect on training time and accuracy.

#### Week 7: Recurrent Neural Networks (RNNs) and LSTMs

* \*\*Lecture Notes:\*\*
* \*\*PDF/Slides:\*"Sequential Data Challenges and the Need for RNNs."
* \*\*PDF/Slides:\*"Recurrent Neural Networks (RNNs): Architecture and Unrolling."
* \*\*PDF/Slides:\*"The Vanishing/Exploding Gradient Problem in RNNs."
* \*\*PDF/Slides:\*"Long Short-Term Memory (LSTM) Networks: Gate Mechanisms."
* \*\*PDF/Slides:\*"Gated Recurrent Units (GRUs) and Bidirectional RNNs."
* \*\*PDF/Slides:\*"Sequence-to-Sequence (Seq2Seq) Models: Encoder-Decoder Intuition."
* \*\*Reading:\*"Understanding LSTM Networks" by Christopher Olah (Blog Post).
* \*\*Code Examples (Python Notebooks/Scripts):\*\*
* \*\*`01\_keras\_simple\_rnn.ipynb`\*\*: Building a basic RNN model for text classification in Keras/TensorFlow.
* \*\*`02\_keras\_lstm\_text\_classification.ipynb`\*\*: Implementing an LSTM model for the same text classification task.
* \*\*`03\_bidirectional\_lstm.ipynb`\*\*: Demonstrating how to use a Bidirectional LSTM layer.
* \*\*`04\_simple\_text\_generation\_lstm.ipynb`\*\*: A basic example of text generation using an LSTM model (e.g., predicting next character/word).
* \*\*Case Studies/Practical Exercises:\*\*
* \*\*Project: Sequence Tagging (Simplified POS Tagging/NER):\*\*
* \*\*Dataset:\*Small dataset of sentences with pre-tagged words (e.g., simplified POS tags or named entities like 'Person', 'Location').
* \*\*Task:\*Train an LSTM model to predict the tag for each word in a sequence.
* \*\*Steps:\*\*

1. Prepare sequence data (padding, numerical representation).

2. Build an LSTM model with an embedding layer.

3. Train and evaluate the model on sequence accuracy.

* \*\*Exercise:\*Train a simple character-level LSTM on a small corpus of text (e.g., a few paragraphs from a book) and generate new text. Observe how it learns patterns.

#### Week 8: Attention Mechanisms & Transformers (Introduction) + Ethical NLP

* \*\*Lecture Notes:\*\*
* \*\*PDF/Slides:\*"Limitations of Seq2Seq without Attention and the Rise of Attention."
* \*\*PDF/Slides:\*"Attention Mechanism: Intuition and Benefits."
* \*\*PDF/Slides:\*"Introduction to Transformers: Self-Attention and Multi-Head Attention."
* \*\*PDF/Slides:\*"Transformer Architecture (Encoder-Decoder High-Level)."
* \*\*PDF/Slides:\*"State-of-the-Art: BERT, GPT, and Pre-training/Fine-tuning."
* \*\*PDF/Slides:\*"Key NLP Applications: MT, Summarization, QA, Chatbots."
* \*\*PDF/Slides:\*"Ethical Considerations in NLP: Bias, Fairness, Privacy, Explainability, Misuse."
* \*\*Reading:\*"Attention Is All You Need" (Vaswani et al.) - high-level overview.
* \*\*Reading:\*Articles/Blog Posts on BERT, GPT-3 (e.g., from Hugging Face, Google AI Blog).
* \*\*Resource:\*AI Ethics Guidelines (e.g., from Google, Microsoft, Partnership on AI).
* \*\*Code Examples (Python Notebooks/Scripts):\*\*
* \*\*`01\_conceptual\_attention\_visualization.ipynb`\*\*: (Optional) A simplified visualization of attention weights on a toy example.
* \*\*`02\_huggingface\_transformers\_intro.ipynb`\*\*: Demonstrates loading and using a pre-trained BERT or GPT-2 model for basic inference tasks (e.g., sentiment analysis, text generation).
* \*\*`03\_fine\_tuning\_bert\_concept.ipynb`\*\*: High-level conceptual code showing how to fine-tune a pre-trained Transformer model for a specific task using `transformers` library's `pipeline` or `Trainer` class (without full training).
* \*\*Case Studies/Practical Exercises:\*\*
* \*\*Exploration: State-of-the-Art NLP Applications:\*\*
* Use Hugging Face's `pipeline` or `model.generate()` to experiment with a pre-trained Transformer for:
* Text Summarization (summarize a news article).
* Question Answering (ask questions about a provided paragraph).
* Text Generation (generate creative text based on a prompt).
* Machine Translation (translate a sentence).
* \*\*Discussion: Ethical NLP Dilemmas:\*\*
* \*\*Case Studies:\*Present specific examples of real-world ethical issues in NLP (e.g., gender bias in translation, racial bias in facial recognition/sentiment analysis, deepfakes, privacy concerns with large language models).
* \*\*Group Activity:\*Students discuss potential solutions, mitigation strategies, and the societal impact of these technologies.
* \*\*Research Task:\*Students research a current trend in NLP (e.g., multimodal NLP, explainable AI for NLP, low-resource NLP) and prepare a short presentation or report.

## Assessments

# Natural Language Processing (NLP) - Assessment Plan

This assessment plan for the 8-week 'Natural Language Processing' undergraduate curriculum is designed to comprehensively evaluate students' understanding of foundational concepts, practical application of techniques, and critical thinking in the field. It incorporates a variety of assessment methods to cater to different learning styles and ensure a holistic evaluation of learning outcomes.

### Overall Grade Breakdown

| Assessment Component | Weight |

| :------------------- | :----- |

| Quizzes | 15% |

| Assignments | 35% |

| Mid-Term Exam | 20% |

| Final Project | 30% |

| **Total** | **100%** |

---

### Assessment Component Details

#### 1. Quizzes (15% of Final Grade)

* \*\*Purpose:\*To regularly assess comprehension of core concepts, terminology, and basic principles introduced in lectures and readings. They encourage consistent engagement with the module material.
* \*\*Frequency:\*Approximately 6 short quizzes, typically administered at the beginning of the week, covering material from the previous week(s).
* \*\*Format:\*Short online quizzes consisting of a mix of multiple-choice questions, true/false, and short answer questions. They are designed to be low-stakes, formative assessments with a summative component.
* \*\*Topics Covered:\*\*
* \*\*Quiz 1 (End of Week 1/Early Week 2):\*Introduction to NLP, Text Preprocessing, Regular Expressions, NLTK/spaCy basics.
* \*\*Quiz 2 (End of Week 2/Early Week 3):\*Language Modeling (N-grams), Perplexity, BoW, TF-IDF.
* \*\*Quiz 3 (End of Week 3/Early Week 4):\*POS Tagging, Introduction to Word Embeddings (Word2Vec, GloVe).
* \*\*Quiz 4 (End of Week 5/Early Week 6):\*Traditional Machine Learning for NLP (Naive Bayes, SVMs), Feature Engineering, Evaluation Metrics (Precision, Recall, F1-score).
* \*\*Quiz 5 (End of Week 6/Early Week 7):\*Neural Network Basics (MLP, Activation Functions, Loss, Optimization).
* \*\*Quiz 6 (End of Week 7/Early Week 8):\*RNNs, LSTMs, GRUs, Sequence-to-Sequence models.

#### 2. Assignments (35% of Final Grade)

* \*\*Purpose:\*To provide hands-on experience in implementing NLP techniques using Python and relevant libraries. These assignments require students to apply theoretical knowledge to practical problems, develop coding skills, and troubleshoot.
* \*\*Frequency:\*3 major programming assignments throughout the course.
* \*\*Format:\*Jupyter Notebooks or Python scripts, submitted via a learning management system, often accompanied by a brief report or README file.
* \*\*Topics & Learning Outcomes Assessed:\*\*
* \*\*Assignment 1: Text Processing & Language Modeling (10%)\*\*
* \*\*Due:\*End of Week 2
* \*\*Description:\*Implement various text preprocessing steps (tokenization, stemming, lemmatization, stop word removal). Build a simple N-gram language model and calculate perplexity. Apply Bag-of-Words and TF-IDF for text representation.
* \*\*Assesses:\*Learning Outcomes from Week 1 & 2.
* \*\*Assignment 2: Word Embeddings & Traditional Classification (15%)\*\*
* \*\*Due:\*End of Week 5
* \*\*Description:\*Utilize pre-trained word embeddings (e.g., GloVe) for a text task. Implement and evaluate a traditional machine learning classifier (e.g., Naive Bayes or SVM) for a text classification problem (e.g., sentiment analysis or spam detection), including feature engineering and evaluation metrics.
* \*\*Assesses:\*Learning Outcomes from Week 3, 4 (conceptual understanding of parsing might inform features), & 5.
* \*\*Assignment 3: Introduction to Neural Networks for Text (10%)\*\*
* \*\*Due:\*End of Week 7
* \*\*Description:\*Build and train a simple feedforward neural network (MLP) or a basic RNN/LSTM model using a deep learning framework (e.g., Keras/TensorFlow or PyTorch) for a text classification or sequence labeling task (e.g., simple POS tagging).
* \*\*Assesses:\*Learning Outcomes from Week 6 & 7.

#### 3. Mid-Term Exam (20% of Final Grade)

* \*\*Purpose:\*A comprehensive assessment of theoretical understanding and conceptual grasp of the first half of the course material.
* \*\*Timing:\*Administered during Week 5, covering material from Week 1 to Week 4.
* \*\*Format:\*A timed, closed-book exam consisting of:
* Definitions and explanations of key NLP concepts.
* Problem-solving questions (e.g., calculating N-gram probabilities, interpreting TF-IDF values, drawing simple parse trees, explaining HMM intuition).
* Short answer questions requiring comparison or critique of different techniques.
* \*\*Topics Covered:\*All topics from Module 1 (Foundations) and Module 2 (Traditional NLP Models) up to Week 4, including: Text Preprocessing, Language Modeling, N-grams, Perplexity, BoW, TF-IDF, POS Tagging, Word Embeddings, Syntactic Parsing (Constituency & Dependency).

#### 4. Final Project (30% of Final Grade)

* \*\*Purpose:\*The capstone assessment, requiring students to integrate knowledge and skills acquired throughout the entire course to solve a significant NLP problem. It fosters independent research, problem-solving, and critical evaluation.
* \*\*Timing:\*Project proposal due Week 5. Final project submission (code and report) due at the end of Week 8.
* \*\*Format:\*Students will work individually or in small groups (2-3 students) on a self-selected or instructor-approved NLP project. Deliverables include:
* \*\*Project Proposal (5% of project grade, part of final project score):\*A brief document outlining the chosen problem, proposed methodology, data sources, and expected outcomes.
* \*\*Code Repository (e.g., GitHub):\*Well-documented, reproducible code implementing the NLP solution.
* \*\*Technical Report (20% of project grade):\*A comprehensive report detailing:
* Problem definition and motivation.
* Literature review (brief).
* Data collection and preprocessing.
* Methodology (models used, experimental setup).
* Results and analysis (including appropriate evaluation metrics).
* Discussion of limitations and future work.
* \*\*Crucially, a dedicated section on ethical considerations\*related to the project's data, models, and potential applications (addressing Week 8's ethical NLP topics).
* \*\*Optional Presentation/Demo (5% of project grade):\*A short presentation or demonstration of the project's functionality and key findings (if time permits or for larger classes, a subset of projects may be presented).
* \*\*Topics & Learning Outcomes Assessed:\*All learning outcomes from the entire 8-week curriculum, with a strong emphasis on applying modern NLP techniques (including deep learning where appropriate) and critically evaluating their societal impact. Examples include advanced sentiment analysis, text summarization, question answering, chatbot development, or bias detection.

## Resources

* Books:
* A Beginner's Guide to Natural Language Processing
* Videos:
* Introduction to Natural Language Processing on Khan Academy
* Websites:
* Official documentation website for Natural Language Processing