NLP Crashcourse

# Table of Contents

1. What is NLP?

2. Approaches to NLP

2.1 Rule-based (Old School Method)

2.2 Statistical Approach (Machine Learning-based)

2.3 Deep Learning Approach (Modern Approach)

3. NLP Pipeline

4. Important Concepts in NLP

4.1 Corpus

4.2 Sentence

4.3 Token/Tokenization

4.4 Dictionaries

5. Feature Extraction & Vectorization

5.1 One Hot Encoding

5.2 Bag of Words

5.3 TF-IDF

5.4 Word2Vec

5.5 N-Grams (Uni-gram, Bi-gram, Tri-gram)

6. Project: Sentiment Analysis (End-to-End Example)

# 1. What is NLP?

Natural Language Processing (NLP) is a subfield of Artificial Intelligence focused on enabling machines to understand, interpret, and generate human language. NLP combines linguistics, computer science, and machine learning to allow computers to read, listen, and respond meaningfully. Everyday examples include chatbots, translation systems, search engines, and sentiment analysis tools.

# 2. Approaches to NLP

## 2.1 Rule-based (Old School Method)

In the rule-based approach, linguistic experts define explicit grammar rules and dictionaries to process text. This method was popular before machine learning became mainstream.

Example: A chatbot that responds using IF-ELSE rules based on keywords.

Advantages:  
- Transparent logic  
- Works well for small domains

Disadvantages:  
- Requires manual rule creation  
- Fails with complex or ambiguous language

Limitations: Cannot scale to real-world large datasets.

## 2.2 Statistical Approach (Machine Learning-based)

This approach uses statistical models and machine learning algorithms. The text is first converted into numerical vectors using methods like Bag of Words or TF-IDF. Then, algorithms like Naïve Bayes, Logistic Regression, or SVM are applied.

Advantages:  
- Learns patterns from data  
- Requires less manual work

Disadvantages:  
- Needs labeled data  
- Performance depends on feature quality

## 2.3 Deep Learning Approach (Modern Approach)

Deep learning revolutionized NLP by using neural networks that automatically learn representations. Architectures include RNNs, LSTMs, GRUs, and Transformers. Modern NLP relies heavily on Transformer-based models such as BERT and GPT.

Advantages:  
- State-of-the-art performance  
- Handles context well

Disadvantages:  
- Requires large data and compute  
- Less interpretable

# 3. NLP Pipeline

1. Data Collection: via scraping, APIs, or manual methods.

2. Text Cleaning: lowercasing, removing punctuation, stopwords, HTML tags, etc.

3. Tokenization: splitting text into words or subwords.

4. Feature Extraction: converting text into numerical form.

5. Model Training: applying ML/DL models.

6. Evaluation and Deployment.

A diagram of a diagram

AI-generated content may be incorrect.

### 4. Important Concepts in NLP (Expanded)

**Corpus**  
A corpus is a large and structured collection of text used to train and evaluate NLP models. It can include books, news articles, tweets, or even transcribed conversations. Corpora can be:

* General-purpose: e.g., Wikipedia, Common Crawl.
* Domain-specific: e.g., PubMed (medical research), legal corpora.  
  The quality, diversity, and size of a corpus directly affect how well an NLP model performs. For instance, a sentiment model trained on movie reviews (IMDB Corpus) may not work well on medical reports.

**Sentence**  
A sentence is a sequence of words expressing a complete thought. Sentences form natural boundaries in language processing. Many NLP tasks, like sentiment analysis or translation, are performed at the sentence level.  
Example:  
Input: "The movie was fantastic, but the ending was disappointing."  
Sentence-level analysis allows the model to detect mixed emotions.

**Tokenization**  
Tokenization breaks text into smaller units called tokens.

* Word-level: "I love NLP" → ["I", "love", "NLP"].
* Subword-level: "unhappiness" → ["un", "happy", "ness"].
* Character-level: "cat" → ["c","a","t"].  
  Modern tokenizers (e.g., BPE, WordPiece) help handle unknown or rare words by splitting them into subwords. Transformers like BERT and GPT rely on these advanced tokenizers.

**Dictionaries**  
In NLP, dictionaries are predefined mappings of words to information such as meaning, sentiment, or numeric IDs.

* Lexical dictionaries: e.g., WordNet for synonyms/antonyms.
* Sentiment dictionaries: "excellent" → positive, "horrible" → negative.
* Index dictionaries: mapping vocabulary to IDs: {"cat":0, "dog":1, "NLP":2}.  
  These dictionaries allow models to process language computationally.

### 5. Feature Extraction & Vectorization

**One Hot Encoding**  
Each word is represented as a binary vector, with 1 at its vocabulary index.

* Example: Vocabulary = ["cat","dog","fish"], word "dog" → [0,1,0].
* Pros: Easy to understand, simple implementation.
* Cons: High dimensionality for large vocabularies, no semantic relationships captured (e.g., "dog" and "puppy" are unrelated).

**Bag of Words (BoW)**  
Represents text by counting word frequencies, ignoring word order.

* Example: "I love NLP, I love AI" → {"I":2,"love":2,"NLP":1,"AI":1}.
* Pros: Works well for classification tasks, easy to implement.
* Cons: Ignores grammar, word order, and context (e.g., "dog bites man" vs "man bites dog" look similar).

**TF-IDF (Term Frequency–Inverse Document Frequency)**  
Improves BoW by weighting words: frequent in one document but rare across corpus = higher weight.

* Example: In a corpus of tech blogs, "algorithm" gets more weight than "the".
* Applications: Document retrieval (search engines), spam detection.
* Pros: Reduces impact of common stopwords.
* Cons: Still ignores deep context and semantics.

**TF-IDF Formula**

The overall formula for the TF-IDF weight of a term (t) in a document (d) from a corpus (D) is:

TF-IDF(t,d,D)=TF(t,d)×IDF(t,D)

**1. Term Frequency (TF)**

**Term Frequency** measures how often a term (t) appears in a document (d).

TF(t,d)=Total number of terms in document dNumber of times term t appears in

document d​.

**2. Inverse Document Frequency (IDF)**

**Inverse Document Frequency** measures how important a term is across the entire corpus (D). It gives a higher weight to terms that are rare across all documents, which is a key improvement over the Bag of Words model.

IDF(t,D)=log(DF(t)N​)

Where:

* N is the **Total number of documents** in the corpus D.
* DF(t) is the **Document Frequency** (the number of documents in D that contain the term t

**Word2Vec**  
A neural embedding model that learns word meanings by context.

* Models: CBOW (predicts a word from its context), Skip-gram (predicts context from a word).
* Famous analogy: King - Man + Woman ≈ Queen.
* Pros: Captures semantics and relationships.
* Cons: Embeddings are static (same vector for "bank" in "river bank" and "money bank").

**N-Grams**  
Sequences of consecutive n words. Captures limited word order.

* Example: "I love NLP"
  + Unigrams: ["I","love","NLP"]
  + Bigrams: ["I love","love NLP"]
  + Trigrams: ["I love NLP"]
* Pros: Simple, preserves some context.
* Cons: Explodes in dimensionality with larger n, sparse for rare phrases.

# 6. Project: Sentiment Analysis

Sentiment Analysis is a common NLP project that classifies text into categories like positive, negative, or neutral. Below is a simplified pipeline:

You can access the project here->

Git hub link: [Sentimental\_Analysis\_Project](https://github.com/Aditya11220/Sentimental-Analysis.git)