Natural Language Processing
NLP_CLT_1c_May_4th_2025

Eng. Maytham Ghanoum

Artificial Intelligence & Deep Learning Specialist

MTN Syria – SCS – SVU CLT

+963947222064 - +963982018359

https://www.linkedin.com/in/maytham-ghanoum-697aa5207/

https://www.facebook.com/maytham.ghanoum

MAYTHAM
GHANOUM
Artificial Intelligence & Deep Learning Specialist

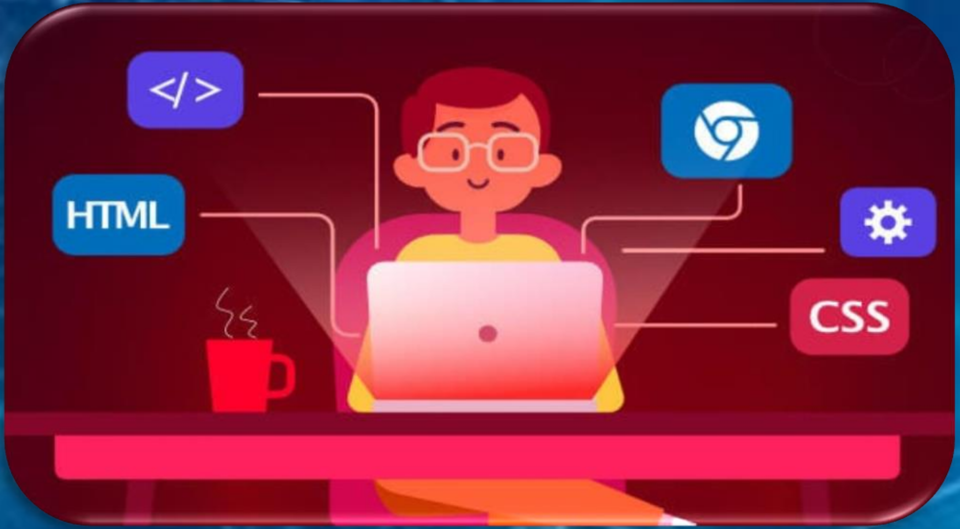
Content

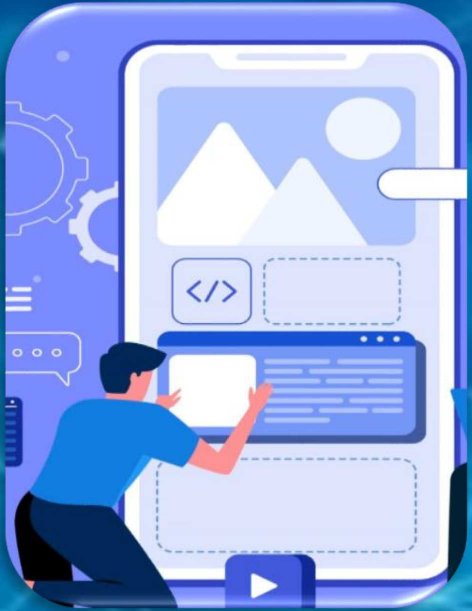
Natural language processing (NLP) is an essential branch of artificial intelligence (AI) that focuses on enabling machines to comprehend, interpret, and respond to human language in a natural and intuitive manner. NLP technology allows machines to understand the meaning behind human language and respond appropriately, enabling a more seamless interaction between humans and machines.

Why we must learn NLP?

- Growing Demand.
- Advanced Language Processing
- Solving Complex Problems
- Multidisciplinary Field
- Lucrative Job Opportunities
- Impactful Applications

Introduction





Who Can use NLP?

Outlines

- NLTK
- Tensorflow & PyTorch
- Basics of Text Processing
- Text Representation
- Deep Learning from scratch
- Text Classification
- Named Entity Recognition (NER)
- Topic Modeling
- Sequence-to-Sequence Models
- Transformers
- Hands-on Projects and Exercises

Tokenization

Sentence Tokenization: This involves splitting text into sentences.

```python
import nltk
from nltk.tokenize import sent_tokenize

# Sample text
text = "NLTK is a leading platform for building
Python programs to work with human language
data."

# Sentence Tokenization
sentences = sent_tokenize(text)
print("Sentence Tokenization:")
print(sentences)
```

Tokenization

Character Tokenization: This involves splitting text into individual characters.

```
import nltk
from nltk.tokenize import sent_tokenize

# Sample text
text = "NLTK is a leading platform for building
Python programs to work with human language
data."

# Character Tokenization
characters = list(text)
print("\nCharacter Tokenization:")
print(characters)
```

Stemming

Stemming is a text normalization technique used in natural language processing (NLP) to reduce words to their root or base form, known as the "stem." The process involves removing suffixes or prefixes from words to obtain the stem. Stemming aims to condense words to their common linguistic root, even if the resulting stem is not a valid word itself. This can help in tasks such as information retrieval, text analysis, and document clustering

Stemming

1.Reducing Dimensionality: Stemming reduces the number of unique words in a corpus by collapsing words with similar meanings into the same stem. This helps in simplifying the vocabulary and reducing the computational complexity of NLP tasks.

2.Improving Text Matching: Stemming enables more accurate matching of words with the same root, even if they are inflected forms or have different prefixes or suffixes. This can improve the performance of tasks such as search engines or document retrieval systems.

Stemming

3. **Preprocessing for Machine Learning:** Stemming is often used as a preprocessing step before applying machine learning algorithms to text data. By reducing words to their stems, the model can focus on the underlying semantic meaning rather than variations in word forms.

4. **Enhancing Interpretability:** Stemming can help in summarizing text data by collapsing related words into a common stem. This can make it easier to interpret the content of documents or analyze trends in a corpus.

Lemmatization

Lemmatization is another text normalization technique used in natural language processing (NLP) to reduce words to their base or dictionary form, known as the "lemma." Unlike stemming, which simply removes suffixes or prefixes to obtain the root form, lemmatization considers the morphological analysis of words and ensures that the resulting lemma is a valid word present in the language's dictionary.

Lemmatization

1. Improved Accuracy: Lemmatization produces valid words as lemmas, which can improve the accuracy of downstream NLP tasks such as text classification, information retrieval, and machine translation.

2. Preservation of Semantic Meaning: Lemmatization preserves the semantic meaning of words by converting them to their base form. This helps in capturing the true meaning of the word and reducing ambiguity in text analysis.

Lemmatization

3.Better Interpretability: Lemmatization results in more interpretable text data by reducing words to their canonical form. This can facilitate better understanding and interpretation of the text by humans and machines alike.

4. Handling of Inflected Forms: Lemmatization handles inflected forms of words by mapping them to their base form in the dictionary. This ensures that variations of the same word are treated consistently in NLP tasks.

Part-of-speech (POS) tagging

Grammatical tagging, is the process of assigning grammatical labels (such as noun, verb, adjective, etc.) to each word in a text corpus based on its syntactic role within a sentence. POS tagging is a fundamental task in natural language processing (NLP) and plays a crucial role in various downstream tasks such as named entity recognition, parsing, sentiment analysis, and machine translation.

Part-of-speech (POS) tagging

1. Syntactic Analysis: POS tagging provides insight into the syntactic structure of sentences by identifying the grammatical roles of words. This information can help in understanding the relationships between words and phrases in a sentence.

2. Semantic Analysis: POS tags can indicate the semantic meaning of words by identifying their syntactic categories. For example, identifying nouns and verbs can help in understanding the subject-action-object relationships in sentences.

Part-of-speech (POS) tagging

3. Ambiguity Resolution: POS tagging helps in disambiguating words with multiple meanings based on their syntactic context. For instance, the word "bank" can be a noun (e.g., river bank) or a verb (e.g., to bank money), and POS tagging helps in determining its correct part of speech.

4. Improving Accuracy of NLP Tasks: Many NLP tasks benefit from POS tagging as it provides useful linguistic information that can improve the accuracy of downstream tasks such as named entity recognition, sentiment analysis, and machine translation..

Part-of-speech (POS) tagging

POS tags in the output:
NNP: Proper noun, singular - This tag indicates that the word is a proper noun (e.g., names of specific people, places, or organizations) and is singular. Examples include "NLTK" and "Python" in the input text.

VBZ: Verb, 3rd person singular present - This tag indicates that the word is a verb in the present tense, third person singular form. Examples include "is" in the input text.

DT: Determiner - This tag indicates that the word is a determiner, which is used to specify nouns. Examples include "a" in the input text.

Syrian Computer Socie

Part-of-speech (POS) tagging

POS tags in the output:
VBG: Verb, gerund or present participle - This tag indicates that the word is a verb in the gerund or present participle form. Examples include "leading" and "building" in the input text.

NN: Noun, singular or mass - This tag indicates that the word is a noun, either singular or mass (uncountable). Examples include "platform" and "language" in the input text.

IN: Preposition or subordinating conjunction - This tag indicates that the word is a preposition or a subordinating conjunction. Examples include "for" and "with" in the input text.

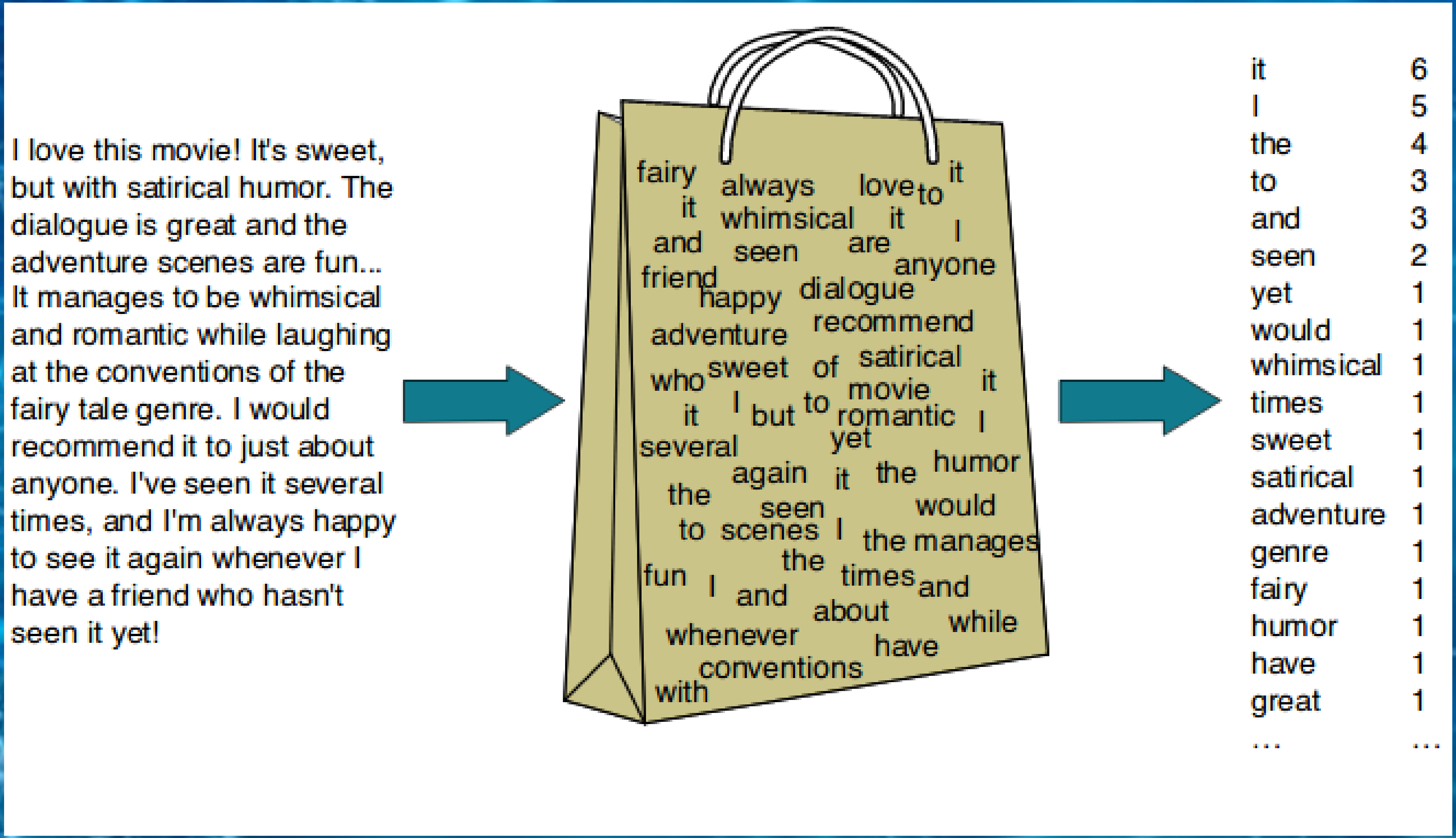
Part-of-speech (POS) tagging

POS tags in the output:
NNS: Noun, plural - This tag indicates that the word is a noun in the plural form. Examples include "programs" and "data" in the input text.

TO: to - This tag indicates the word "to" as part of an infinitive verb form or as a preposition. Example includes "to" in the input text.

JJ: Adjective - This tag indicates that the word is an adjective, which describes or modifies a noun. Example includes "human" in the input text.
. : Punctuation - This tag indicates that the word is punctuation, such as periods, commas, or other symbols. Example includes "." in the input text.

Bag of Words



I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!

it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1
...	...

BOW

The Bag-of-Words (BoW) model is a fundamental method for text representation in Natural Language Processing (NLP). It is a way to represent text data (such as sentences or documents) as numerical vectors, which can be used as input for various machine learning algorithms.

In the BoW model, a text (e.g., a sentence or a document) is represented as a multiset of its words, disregarding grammar and word order but keeping multiplicity. Here's a step-by-step process to understand how BoW works:

Vocabulary Creation: Compile a list of all unique words (the vocabulary) in the entire text corpus.
Vector Representation: For each document in the corpus, create a vector where each dimension corresponds to a word in the vocabulary. The value of each dimension is the count (or frequency) of the word in the document.

Syrian Computer Socie

BOW

Consider a simple corpus with three sentences:

- "I love machine learning"
- "Machine learning is great"
- "I love coding“

The rule that generates this representation involves the following steps:
1- Create the Vocabulary: Identify all unique words in the entire corpus (the collection of sentences/documents).
Vocabulary: ['I', 'love', 'machine', 'learning', 'is', 'great', 'coding']

Syrian Computer Socie

BOW

2- **Order the Vocabulary:** The order of the vocabulary can be arbitrary but needs to be consistent. For instance, in our example, the order is:

[‘coding’,‘great’, ‘I’, ‘is’ , ‘learning’, ,‘Love’, ‘machine’]

2- Vector Representation: For each document, create a vector where each dimension corresponds to a word in the vocabulary. The value of each dimension is the count (frequency) of the word in the document.

BOW

Detailed Steps for the Sentence "I love machine learning":

Initialize the Vector: Start with a zero vector that has the same length as the vocabulary. In this case, the initial vector is:

[0,0,0,0,0,0,0]

Count Word Frequencies:

"I" appears once: Increment the first position (corresponding to 'I') by 1.

"love" appears once: Increment the second position (corresponding to 'love') by 1.

"machine" appears once: Increment the third position (corresponding to 'machine') by 1.

"learning" appears once: Increment the fourth position (corresponding to 'learning') by 1.

Syrian Computer Socie

BOW

Resulting Vector: After counting the frequencies of the words in the sentence, the vector becomes:

Vector Representation:

"I love machine learning": [1, 1, 1, 1, 0, 0, 0]
"Machine learning is great": [0, 0, 1, 1, 1, 1, 0]
"I love coding": [1, 1, 0, 0, 0, 0, 1]

BOW

Benefits of the Bag-of-Words Model:

Simplicity: BoW is straightforward to understand and implement.

Effective for Small Datasets: It's effective for simple text classification tasks, especially with smaller datasets.

Baseline Model: It serves as a good baseline for more complex models.

BOW

Use Cases of Bag-of-Words in NLP
Text Classification: Spam detection, sentiment analysis, document categorization.

Information Retrieval: Search engines, document similarity.

Language Modeling: Basic language generation tasks, creating simple chatbot responses..