**1. Explain One-Hot Encoding**

**One-Hot Encoding** is a method used to convert categorical data (words or labels) into a binary vector format. Each category is represented as a vector where only one position is **"1"**, and the rest are **"0"**.

**Example:**  
For three words: "cat", "dog", "fish", the One-Hot Encoding representation would be:

| **Word** | **Vector Representation** |
| --- | --- |
| cat | [1, 0, 0] |
| dog | [0, 1, 0] |
| fish | [0, 0, 1] |

**Advantages:** Simple and easy to implement.  
**Disadvantages:** Leads to a **high-dimensional sparse matrix** if the vocabulary is large.

**2. Explain Bag of Words (BoW)**

The **Bag of Words (BoW)** model represents text as a collection of word frequencies, **ignoring grammar and word order**.

**Example:**  
Consider these two sentences:

1. "I love NLP"
2. "NLP is fun"

The vocabulary (unique words) = {I, love, NLP, is, fun}

| **Sentence** | **I** | **love** | **NLP** | **is** | **fun** |
| --- | --- | --- | --- | --- | --- |
| I love NLP | 1 | 1 | 1 | 0 | 0 |
| NLP is fun | 0 | 0 | 1 | 1 | 1 |

**Advantages:** Easy to implement, good for simple tasks like spam detection.  
**Disadvantages:** **Loses context** and word relationships.

**3. Explain Bag of N-Grams**

**Bag of N-Grams** is an extension of **BoW** where instead of single words, **N-word sequences** (N-grams) are considered. It helps preserve some context.

**Example (for bigrams, N=2):**  
Sentence: "I love NLP"

* **Bigrams (2-grams):** "I love", "love NLP"

| **Sentence** | **I love** | **love NLP** |
| --- | --- | --- |
| I love NLP | 1 | 1 |

**Advantages:** Captures **some word order information**.  
**Disadvantages:** Increases **feature space** significantly.

**4. Explain TF-IDF (Term Frequency - Inverse Document Frequency)**

**TF-IDF** is a statistical measure that evaluates how important a word is in a document relative to a collection of documents (corpus).

**Formula:**

* **TF (Term Frequency):** How often a word appears in a document.
* **IDF (Inverse Document Frequency):** Penalizes common words that appear in many documents.

TF−IDF=TF×IDFTF-IDF = TF \times IDFTF−IDF=TF×IDF

**Example:**  
A word appearing **5 times in a document** but is common across all documents will have a **low TF-IDF score**, while a unique word appearing **less frequently** will have a **high TF-IDF score**.

**Advantages:** Helps filter out **common words** while keeping meaningful ones.  
**Disadvantages:** Doesn’t capture **word order or meaning**.

**5. What is the OOV (Out-of-Vocabulary) Problem?**

The **OOV problem** occurs when a model encounters a word it has **never seen during training**.

**Example:**  
A language model trained on **sports articles** may not understand **medical terms**, leading to **poor predictions**.

**Solutions:**

* Use **word embeddings** (Word2Vec, GloVe).
* Apply **subword tokenization** (e.g., Byte-Pair Encoding, SentencePiece).
* Use a **special <UNK> token** for unknown words.

**6. What Are Word Embeddings?**

**Word embeddings** are vector representations of words in a continuous space where **similar words have similar numerical representations**. Unlike One-Hot Encoding, embeddings **capture meaning, relationships, and context**.

**Example:**  
The words **"king"**, **"queen"**, and **"royalty"** will have embeddings close to each other in vector space.

**Popular Models:**

* **Word2Vec (CBOW & Skip-Gram)**
* **GloVe**
* **FastText**

**7. Explain Continuous Bag of Words (CBOW) (Word2Vec Approach)**

**CBOW** predicts a **target word** using its **context words**.

**Example:**  
For the sentence **"I love natural language processing"**, the model is trained to predict "language" given its surrounding words:

Input: ["I", "love", "natural", "processing"] → Output: "language"\text{{Input: ["I", "love", "natural", "processing"] → Output: "language"}}Input: ["I", "love", "natural", "processing"] → Output: "language"

**Advantages:**

* **Efficient training** (better for smaller datasets).
* Works well for **common words**.

**8. Explain Skip-Gram (Word2Vec Approach)**

**Skip-Gram** is the **opposite of CBOW**. It predicts **context words** given a **single target word**.

**Example:**  
For "I love NLP", the model takes "love" as input and predicts its context words: ["I", "NLP"].

**Advantages:**

* Works well for **rare words**.
* Captures **complex word relationships**.

**9. Explain GloVe Embeddings**

**GloVe (Global Vectors for Word Representation)** is a word embedding technique that captures both **local context** (like Word2Vec) and **global corpus-wide word co-occurrence statistics**.

**Example:**

* **Word2Vec**: Learns embeddings from a **neural network**.
* **GloVe**: Uses **matrix factorization** to learn embeddings from a co-occurrence matrix.

**Advantages:**

* **More efficient** than Word2Vec.
* Captures **both meaning and global context**.