Class activity – CBOW

Name: Satyatma Chincholi

USN: 1RVU23CSE414

Date: 23/02/2026

1. CBOW (Continuous Bag of Words)

CBOW is an architecture for Word2Vec where the model predicts a **target word** based on the surrounding **context words**.

- **Logic:** If the sentence is "The cat sits on the mat," and the target is "sits," the context words might be ["The", "cat", "on", "the"].
- **Use Case:** Faster to train than Skip-gram and has slightly better accuracy for frequent words.

Python Code Example (Simplified via Gensim)

```python
from gensim.models import Word2Vec

# Sample sentences
sentences = [["cat", "say", "meow"], ["dog", "say", "woof"]]

# sg=0 selects CBOW (sg=1 would be Skip-gram)
model = Word2Vec(sentences, vector_size=10, window=2, min_count=1, sg=0)

# Get the vector for 'cat'
vector = model.wv['cat']
print(vector)
```

2. Skip-gram

Skip-gram is the inverse of CBOW. It uses a **single target word** to predict the **surrounding context words**.

- **Logic:** For the word "sits," it tries to predict "cat," "on," etc.
- **Use Case:** Works better with a small amount of training data and represents rare words or phrases well.

3. GloVe (Global Vectors for Word Representation)

GloVe is an unsupervised learning algorithm developed by Stanford for generating word embeddings. Unlike Word2Vec, which is predictive (local context), GloVe is based on **Matrix Factorization** (global context).

- **Example:** It looks at a global co-occurrence matrix (how often word X appears near word Y in the *entire* dataset).
- **Intuition:** The relationship between "King" and "Queen" should be mathematically similar to the relationship between "Man" and "Woman."

- $Vector(King) - Vector(Man) + Vector(Woman) \approx Vector(Queen)$

4. Difference: Word2Vec (W2V) vs. GloVe

Feature	Word2Vec (W2V)	GloVe
Model Type	Predictive (Neural Network)	Count-based (Matrix Factorization)
Context	Local (moving window)	Global (entire corpus statistics)
Performance	Better at analogy tasks	Better at word similarity and small datasets
Training	Trains on individual windows	Trains on a pre-computed co-occurrence matrix

5. LeNet-5

Developed by Yann LeCun in 1998, **LeNet** was one of the first successful Convolutional Neural Networks (CNNs). It was specifically designed for handwritten digit recognition (MNIST).

- **Architecture:** It consists of 2 Convolutional layers, 2 Subsampling (Pooling) layers, and 3 Fully Connected layers.
- **Significance:** It introduced the idea of using convolutions to extract spatial features automatically.

6. VGGNet (VGG-16)

VGGNet (from the Visual Geometry Group at Oxford) pushed the boundaries of depth in CNNs.

- **Key Feature:** It uses very small (3×3) convolution filters but stacks many of them to create a deep network (e.g., VGG-16 has 16 weight layers).
- **Why it matters:** It proved that increasing the depth of a network significantly improves performance in image classification.

7. LlamaIndex

LlamaIndex (formerly GPT Index) is a data framework for **Large Language Model (LLM)** applications.

- **Function:** It acts as a bridge between your **private data** (PDFs, databases, APIs) and an **LLM** (like GPT-4 or Llama 3).
- **Use Case:** If you want to build a "Chat with my PDF" tool, LlamaIndex handles the indexing of that PDF so the LLM can search it and answer questions accurately.
- **Mechanism:** It uses **RAG (Retrieval-Augmented Generation)** to fetch relevant context before sending a prompt to the AI.