Unit V Computational Intelligence and NLP

5.1 Introduction, Word embedding Techniques-Bag of Words, TF-IDF,Word2Vec, Glove

What are Word Embedding Techniques?

Word embedding techniques are ways to represent words as **numbers** (vectors) so that computers can understand and process text. These methods help to capture the **meaning** of words based on the context in which they appear.

---

1. Bag of Words (BoW)

- **How It Works**:
  - It represents text by counting how often each word appears.
  - The order of words is ignored.
- **Example**: For the two sentences:
  - "I love programming"
  - "Programming is fun"

We count how many times each word appears:

kotlin

CopyEdit

[I, love, programming, is, fun]

Document 1: [1, 1, 1, 0, 0] (1 means the word appears) Document 2: [0, 0, 1, 1, 1]

- **Pros**: Simple and easy to use.
- **Cons**: Ignores word order and context, and can lead to high-dimensional vectors.

---

2. TF-IDF (Term Frequency-Inverse Document Frequency)

- **How It Works**:
    - **TF**: Measures how often a word appears in a document.
    - **IDF**: Measures how rare or common a word is across all documents.
    - Words that are common in a document but rare in the entire dataset are given higher importance.
- **Example**: Words like "the" or "is" are common and get low scores, while unique words like "programming" get higher scores.
- **Pros**: Helps find important words by considering their rarity across documents.
- **Cons**: Still doesn't capture meaning or context.

---

3. Word2Vec

- **How It Works**:
    - It uses a neural network to learn word representations based on the words around them in sentences.
    - Words with similar meanings will have similar vector representations.
- **Example**: Words like "dog" and "puppy" will have similar vectors because they are closely related in meaning.
- **Pros**: Captures word meanings based on context (e.g., "king" and "queen" are related).
- **Cons**: Needs a lot of data to work well and may struggle with rare words.

---

4. GloVe (Global Vectors for Word Representation)

- **How It Works**:
    - It uses statistics from the entire corpus (all documents) to find how often words appear together.

  - Words that frequently appear together will have similar vectors.
- **Example**: "Apple" and "fruit" will have similar vectors because they often appear together.
- **Pros**: Captures global relationships between words across the entire dataset.
- **Cons**: Can be slow and requires a lot of data.

Quick Comparison:

Technique	What It Does	Pros	Cons
Bag of Words	Counts word frequency.	Simple, easy to implement.	Ignores word order and meaning.
TF-IDF	Adjusts word frequency by how rare it is.	Highlights important words.	Doesn't capture word meaning.
Word2Vec	Learns word meanings based on context.	Captures word meanings.	Needs a lot of data.
GloVe	Uses global statistics to capture word meaning.	Captures relationships between words.	Can be slow and data-heavy.

5.2 Neural word embedding, Neural Machine Translation, Seq2Seq and Neural Machine Translation, translation Metrics (BLEU Score & BERT Score)

1. Neural Word Embedding

What is Neural Word Embedding?

Neural word embedding is a technique that represents words as **dense vectors** (arrays of numbers) using **neural networks**. Unlike traditional methods like **Bag of Words**, which treat words as separate entities, neural word embeddings capture the **semantic relationships** between words based on their context.

- **How It Works**:

    - Neural networks are trained on large text data to learn **word representations**.
    - Similar words (like "cat" and "dog") are represented by vectors that are closer to each other in the vector space.
- **Example**: After training, the words "king" and "queen" will have similar vector representations because they share similar meanings and relationships (like "king" - "man" ≈ "queen" - "woman").

---

2. Neural Machine Translation (NMT)

What is Neural Machine Translation (NMT)?

Neural Machine Translation (NMT) is a method of translating text from one language to another using **neural networks**. NMT uses large datasets of text in multiple languages to train a model that can learn the mapping between source and target languages.

- **How It Works**:
    - The model is trained on pairs of sentences in the source and target languages (e.g., "I love programming" → "Ich liebe Programmieren").
    - The neural network learns to **generate** the target language sentence from the input sentence.
- **Advantages of NMT**:
    - More **accurate** and **natural** translations compared to older rule-based systems.
    - Captures **contextual meaning**, leading to better translations.

---

3. Seq2Seq (Sequence to Sequence) and NMT

What is Seq2Seq?

Seq2Seq (Sequence to Sequence) is a deep learning model used for tasks like **machine translation**. It consists of two parts:

1. **Encoder**: Takes an input sequence (e.g., a sentence in the source language) and converts it into a fixed-length vector (called the **context vector**).
2. **Decoder**: Takes the context vector and generates an output sequence (e.g., a translation in the target language).

- **How It Works in NMT**:
  - In NMT, the **encoder** processes the source sentence (e.g., "I love programming"), and the **decoder** generates the translated sentence (e.g., "Ich liebe Programmieren").
  - **Attention mechanisms** (a more advanced technique) can improve Seq2Seq models by allowing the decoder to focus on different parts of the input sentence as it generates each word in the output.
- **Advantages**:
  - Works well for tasks where both the input and output are sequences (like translation, summarization, etc.).
  - **Attention mechanisms** help the model focus on the important words in the input sequence.

---

4. Translation Metrics (BLEU Score & BERT Score)

What are Translation Metrics?

Translation metrics are used to **evaluate the quality** of machine-generated translations. They compare the output of the machine translation model with a **reference translation** to determine how accurate the translation is.

1. BLEU Score (Bilingual Evaluation Understudy)

- **What is BLEU?**

- **BLEU** is a metric used to evaluate how well a machine translation model performs by comparing the machine-generated translation with human-produced reference translations.
- It calculates the **precision** of n-grams (sequences of 1, 2, 3, etc., words) in the predicted translation compared to the reference translations.

- **How It Works**:
  - **Precision**: Measures how many n-grams in the generated translation appear in the reference translations.
  - **Brevity Penalty**: If the translation is shorter than the reference translation, the BLEU score is penalized to avoid generating overly short translations.
- **Example**:
  - Reference: "I love programming."
  - Machine Translation: "I enjoy programming."
  - BLEU will compare the n-grams (e.g., "I love", "programming") and give a score based on overlap.
- **Pros**: Fast, easy to compute.
- **Cons**: Doesn't account for word order or synonyms well.

2. BERT Score

- **What is BERT Score?**
  - **BERT Score** is a newer metric that uses **BERT**, a powerful language model, to evaluate translations.
  - It measures the **semantic similarity** between the machine-generated translation and the reference translation using contextual embeddings from BERT.
- **How It Works**:

- Instead of comparing exact matches like BLEU, BERT Score looks at the **contextual meaning** of words in the translation.
- It uses **precision, recall, and F1 score** to evaluate the quality of the translation.

- **Advantages**:
  - Takes **context** into account, making it more accurate than traditional metrics like BLEU.
  - Better at evaluating the quality of translations in cases where synonyms or word order differs.

Summary of Concepts

Concept	Description
Neural Word Embedding	Represents words as vectors based on context to capture meanings.
Neural Machine Translation (NMT)	Uses neural networks to translate text between languages.
Seq2Seq	A model for translating sequences (like sentences) using an encoder-decoder structure.
BLEU Score	A metric for evaluating machine translation based on n-gram overlap with reference translations.
BERT Score	A metric for translation quality based on semantic meaning, using BERT embeddings.

5.3 Traditional Versus Neural Metrics for Machine Translation Evaluation, Neural **Style Transfer, Pertained NLP BERT Model and its application**

1. Traditional vs. Neural Metrics for Machine Translation Evaluation

Traditional Metrics (e.g., BLEU)

- **What They Are**: Traditional metrics like **BLEU** (Bilingual Evaluation Understudy) are used to evaluate machine translation by comparing the machine-generated translation to reference translations created by humans.
- **How They Work**: These metrics mainly focus on the **exact match** of n-grams (sequences of words) between the machine translation and the reference translation. The goal is to see how many words or phrases overlap with the reference translation.
- **Examples**:
  - **BLEU**: Measures precision of n-grams in the machine-generated translation compared to reference translations. It uses a **brevity penalty** to penalize translations that are too short.
- **Pros**: Fast and computationally cheap. Easy to implement.
- **Cons**: Doesn't account for word order, synonyms, or semantic meaning. It only cares about surface-level matches (exact words or phrases).

Neural Metrics (e.g., BERT Score)

- **What They Are**: Neural metrics use **neural networks**, particularly models like **BERT**, to evaluate translation quality. Instead of comparing exact words, they look at the **semantic meaning** of the translation.
- **How They Work**: These metrics leverage the power of **contextual embeddings** produced by neural models (like BERT) to measure the **similarity** between the machine-generated translation and the reference translation. They focus on understanding the meaning of the words in context, rather than their exact match.
- **Examples**:
  - **BERT Score**: Measures the semantic similarity between the machine translation and reference translation based on BERT's word embeddings. It evaluates the translation's **precision**, **recall**, and **F1 score** based on contextual meanings.

- **Pros**: More accurate than traditional metrics as they consider word meanings and context. Can handle synonyms and word order differences.
- **Cons**: Computationally expensive and slower than traditional metrics. Requires access to powerful models like BERT.

Summary: Traditional vs. Neural Metrics

Type	**Traditional Metrics (e.g., BLEU)**	**Neural Metrics (e.g., BERT Score)**
Focus	Exact word match (n-grams)	Semantic meaning, context
Evaluation Method	Measures n-gram overlap	Measures similarity based on embeddings
Pros	Fast, easy to compute	More accurate, understands context
Cons	Ignores synonyms, word order	Computationally expensive, slower

---

2. Neural Style Transfer

What is Neural Style Transfer?

Neural Style Transfer (NST) is a deep learning technique that allows us to combine the **content** of one image with the **style** of another. This creates an output image that **looks like the first image** but has the artistic style of the second image (e.g., turning a photograph into a painting).

- **How It Works**:
  - **Content Image**: The image whose **structure and content** you want to preserve (e.g., a photo of a person).

- **Style Image**: The image whose **artistic style** you want to apply (e.g., a painting by Van Gogh).
- **Model**: A convolutional neural network (CNN) is used to extract features (like shapes and textures) from both images. The model combines the content features of the first image with the style features of the second image.

- **Result**: The final image will **look like the content image**, but with the **artistic style** of the style image.
- **Example**:
  - Input: A photo of a city.
  - Style: A Van Gogh painting.
  - Output: The photo of the city transformed into a Van Gogh-style painting.
- **Applications**:
  - Artistic image generation.
  - Style-based photo editing.
  - Content generation for creative industries.

---

3. Pretrained NLP BERT Model and its Applications

What is BERT?

BERT (Bidirectional Encoder Representations from Transformers) is a state-of-the-art pretrained language model developed by Google. It's based on **Transformers**, a deep learning architecture that can process text in a **bidirectional** manner (understanding the context of a word based on both the words before and after it).

- **How It Works**:
  - **Pretraining**: BERT is trained on a large corpus of text (like Wikipedia) to learn patterns in the language.

- **Fine-tuning**: Once pretrained, BERT can be **fine-tuned** on specific tasks, such as question answering, sentiment analysis, and named entity recognition (NER).

- **Advantages of BERT**:
  - **Bidirectional**: It looks at the context before and after a word, which allows it to better understand meanings.
  - **Pretrained**: BERT has already been trained on vast amounts of data, so you don't have to train it from scratch. You can fine-tune it for specific tasks.

Applications of BERT:

1. **Sentiment Analysis**:
   - **Task**: Understanding whether a text expresses a positive or negative sentiment (e.g., analyzing customer reviews).
   - **BERT Application**: BERT can understand the sentiment of a text by processing the context of the words in the sentence.
2. **Named Entity Recognition (NER)**:
   - **Task**: Identifying entities like names, dates, and locations in text.
   - **BERT Application**: BERT can identify these entities by understanding the relationships between words.
3. **Question Answering**:
   - **Task**: Answering questions based on a passage of text.
   - **BERT Application**: BERT can read the passage and answer the question by understanding the context of both the question and the passage.
4. **Text Classification**:
   - **Task**: Categorizing text into different topics or labels (e.g., spam detection in emails).

- **BERT Application**: BERT can classify text by understanding the underlying meaning of the text.

Summary of BERT's Applications

Application	Description
Sentiment Analysis	Determines if a text is positive or negative.
Named Entity Recognition	Identifies specific entities like names, dates.
Question Answering	Answers questions based on a given text.
Text Classification	Categorizes text into labels (e.g., spam vs. not spam).