Minor in AI

Predicting the Future

Mastering Next Word Prediction in NLP

March 24, 2025

1 Overview of Neural Networks

Neural networks are powerful function approximators that learn mappings between inputs and outputs. Depending on the data structure, different architectures are preferred:

- **Multilayer Perceptrons (MLPs)**: Suitable for structured, tabular data.
- **Convolutional Neural Networks (CNNs)**: Best for image processing tasks.
- **Recurrent Neural Networks (RNNs)**: Designed for sequential data processing.
- **GRUs and LSTMs**: Address limitations of simple RNNs by handling long-term dependencies.
- **Bidirectional RNNs**: Improve context understanding in sequential data.

2 Embedding Techniques

One-hot vectors can be inefficient due to sparsity. Instead, dense vector representations (embeddings) are used. Methods include:

- **Word2Vec**
- **GloVe**

These embeddings improve the ability of models to capture relationships between words.

3 Applications in NLP

3.1 Sentiment Analysis

A common NLP application where sequences (e.g., sentences) are classified into sentiment categories (positive, neutral, negative). A **Many-to-One** architecture is commonly used.

Figure 1: Sentiment Analysis

3.2 Sequence-to-Sequence Models

Used in applications like machine translation, where both input and output are sequences. *Google Translate* is a well-known example, utilizing attention mechanisms to improve translation quality.

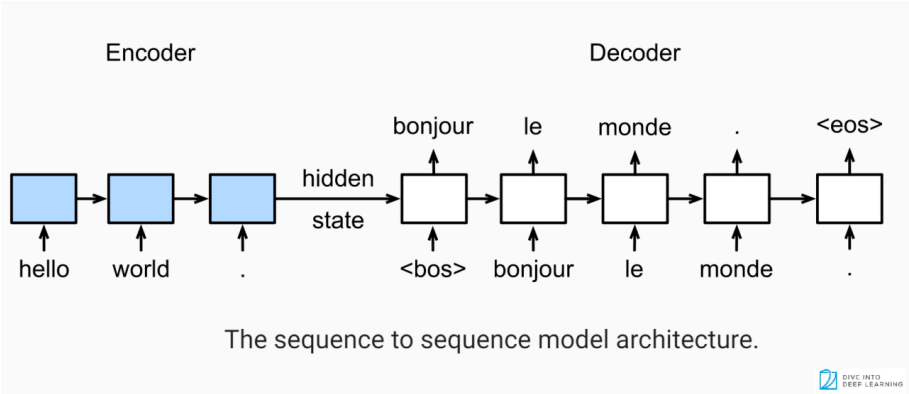

Figure 2: Sequence to Sequence Modeling P.C : LinkedIn

4 Attention Mechanism

Attention allows a model to focus on relevant parts of the input at different time steps, overcoming limitations of fixed-length encoding in RNNs and LSTMs.

Attention mechanisms help models focus on different parts of the input sequence when generating output. Traditional sequence models like RNNs and LSTMs process information sequentially, meaning earlier words might be forgotten as the sequence gets longer. Attention solves this issue by assigning different importance (weights) to different words in the input at each step of the output generation.

4.1 How Attention Works

- Each input word gets a weight representing its relevance to the current output word.
- These weights are calculated using a scoring function.
- The model then takes a weighted sum of input representations to create a context vector.
- This context vector is used to generate the output.

4.2 Example: Translating "`The cat is on the mat`"

In translation, attention allows the model to focus on relevant words from the source sentence while generating each word in the target sentence.

For instance, when translating "`cat`", the attention mechanism ensures the model assigns higher importance to "`cat`" in the input sentence rather than unrelated words like "`the`".

For a more concrete example, consider translating from *English* to *French*:

- Input sentence: `"The cat is on the mat."`
- Target sentence: `"Le chat est sur le tapis."`

When generating "`chat`", the attention mechanism assigns high weights to "`cat`" in the input. When generating "`tapis`", the model focuses on "`mat`".

This dynamic weighting improves translation accuracy.

4.3 Application in Transformers

The attention mechanism is crucial in modern NLP architectures like the Transformer model, which powers state-of-the-art translation systems like Google Translate. It allows parallel processing and better long-range dependencies compared to traditional RNN-based models.

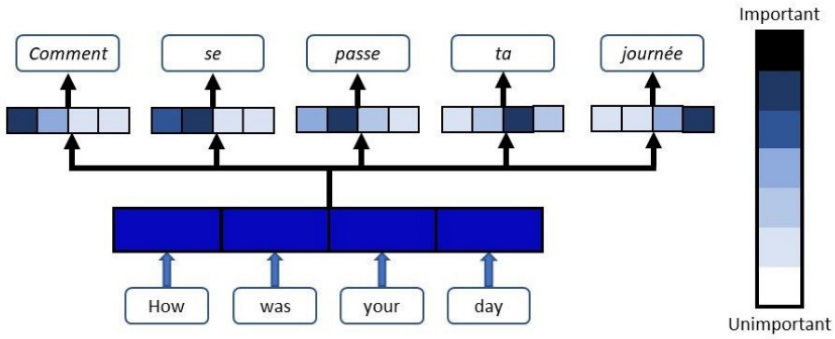

Figure 3: Attention Mechanism P.C : Teksands.AI

5 Model Evaluation: BLEU Score

The Bilingual Evaluation Understudy (BLEU) score measures the quality of generated text against human references. It is based on n-gram precision with a brevity penalty.

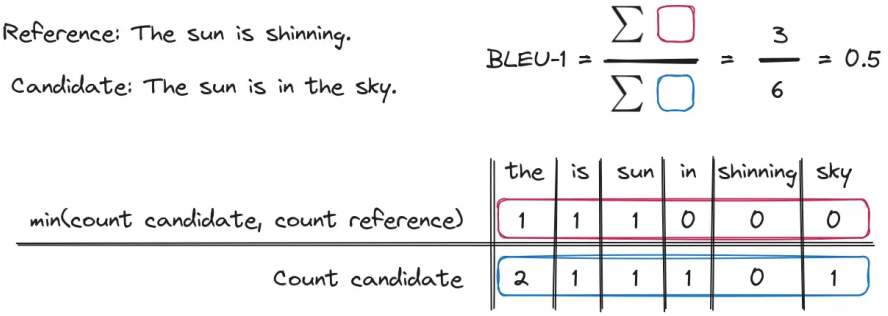

Figure 4: BLEU Score P.C.: Elastic

5.1 Unigram Precision Calculation

Consider the following candidate and reference translations:

Word	Candidate Count	Reference Count
the	2	1
cat	1	1
is	1	1
on	1	1
mat	1	1

The matched unigrams are 5, and the total candidate unigrams are 6. Thus, the unigram precision P_1 is:

$$P_1 = \frac{5}{6} \approx 0.83 \tag{1}$$

5.2 Bigram Precision Calculation

If the bigrams in the candidate are:

`the cat`, `cat is`, `is on`, `on mat`

and in the reference:

`the cat`, `cat is`, `is on`, `on the`, `the mat`

The matched bigrams are 3 out of 4 candidate bigrams:

$$P_2 = \frac{3}{4} = 0.75 \tag{2}$$

6 Brevity Penalty (BP)

To penalize short translations, BLEU includes a brevity penalty BP:

$$BP = e^{(1-\frac{r}{c})} \tag{3}$$

where r is the reference length and c is the candidate length.

6.1 Example: Calculating Brevity Penalty

Consider an example where,

- Reference translation length $r = 10$
- Candidate translation length $c = 8$

Applying the brevity penalty formula:

$$BP = e^{(1-\frac{10}{8})} = e^{-0.25} \approx 0.7788 \tag{4}$$

Since the candidate translation is shorter than the reference, the BLEU score will be penalized by multiplying with this BP factor.

7 Key Takeaways

1. Different neural network architectures are suited for different data types.
2. Embeddings improve representation learning for NLP tasks.
3. Sequence-to-sequence models and attention mechanisms enhance translation models.
4. BLEU score evaluates text generation quality using n-gram precision and brevity penalty.