READ MEDIUM
Search
Translate to English

Tiroshan Madushanka

Summary

The web content discusses the concept of word embeddings in Natural

Use the OpenAI 01 models for free at OpenAI01.net (10 times a day for free)!

Word Embedding: Understanding the Fundamentals

READ MEDIUM
Translate to

In the journey of learning about AI, we have previously discussed the fundamental component used in machine learning techniques, vectors, in a previous article in the series "Zero to AI." Today, we will dive into another critical component in Natural Language Processing (NLP), **word embeddings**.

Reasons why we need word embeddings.

- **High-dimensional representation**: Words can have complex relationships, and it is challenging to represent these relationships using traditional one-hot encoding. Word embeddings provide a lower-dimensional representation that can capture the relationships between words.

Word embeddings, on the other hand, provide a continuous and dense representation that can be used to measure similarity and distance between words.

- **Improved model performance:** Most NLP models require numerical inputs, and one-hot encoded vectors are unsuitable for these models. Word embeddings, on the other hand, provide a better representation for NLP models, which can lead to improved performance in tasks such as text classification and sentiment analysis.
- **Transfer learning:** Word embeddings trained on a large corpus can be used as pre-trained models for other NLP tasks, reducing the training data required for a new job.

Word Embedding

Word embedding represents words as vectors in a high-dimensional space, making it easier to perform mathematical operations and perform similarity analysis. The goal of word embeddings is to provide a way of representing words in a way that can be used in models and algorithms. There are different approaches to generating word embeddings, each with pros and cons.

The first approach is one-hot encoding, where each word is represented as a vector with a 1 in the position corresponding to the word and 0s everywhere else. This approach is simple but has several drawbacks. For example,

READ MEDIUM

Translate to

information about what the words mean.

Another approach is hand-engineered rules, where each word is assigned a number based on predefined categories and weights. This approach works for a limited number of words but is not scalable since it requires some knowledge about each word you want to encode.

READ MEDIUM
Translate to

The following approach uses raw term frequency, where a word vector represents the raw count of how many times the word appears in a document. This approach may help distinguish between different documents, but overrepresented words like "the" and "is" and meaningful words that do not show up often will be lost.

A variation of this approach is TF-IDF, or weighted term frequency, where instead of the raw count, the word vector is normalized by the frequency of the word across all documents. This approach may help distinguish between different documents but still needs to provide information about the meaning of the words.

READ MEDIUM
Translate to

Another method is to use context windows, which look at the words around a specific word to understand its meaning. By analyzing co-occurrence or how often words appear in the same context, a matrix can be created that represents the relationships between words. However, this method requires a large corpus of documents and can result in a sparse matrix. To handle this, the matrix can be reduced through dimensionality reduction approaches.

READ MEDIUM
Translate to

Finally, we can utilize the power of neural networks to generate word embeddings. The two most common models used to generate word embeddings are CBOW (Continuous Bag of Words) and SkipGram. CBOW predicts a word given its context window, while SkipGram predicts the next word given the word. SkipGram is the most widely used model and is the basis for the popular word2vec embedding.

READ MEDIUM
Translate to

In SkipGram, a one-hot encoded vector is used to represent a word. This vector is then input into a hidden layer where weights are assigned to it. The model's objective is to predict the probability of every word in a corpus being the next word. The output layer of the model is not what we are interested in; instead, it is the hidden layer that contains the weight matrix, which represents the word embedding.

READ MEDIUM
Translate to

http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/

The hidden layer of the model is a weight matrix with one row per word and one column per neuron. This weight matrix results from forward and backward propagation and gradient descent, which learn the correct weights for each word. This is the condensed representation of the word that preserves its context.

In this series, we will discuss more details on CBOW and Skip-gram.

have the advantage of capturing the semantic meaning of words. Still, there are also some disadvantages, such as the computational cost of training and potential bias towards the corpus they are trained on.

References

http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/

Word Embeddings | Machine Learning | NLP | Artificial Intelligence

Recommended from ReadMedium


Mdabdullahalhasib

A Complete Guide to Embedding For NLP & Generative AI/LLM

Understand the concept of vector embedding, why it is needed, and implementation with LangChain.

11 min read


Harsh Vardhan

referred to as word embeddings—is...

10 min read

Amit Yadav

Langchain vs Huggingface

If you think you need to spend $2,000 on a 120-day program to become a data scientist, then listen to me for a minute.

10 min read

Abdur Rahman

Python is No More The King of Data Science

5 Reasons Why Python is Losing Its Crown

7 min read

Vipra Singh

LLM Architectures Explained: Word Embeddings (Part 2)

Deep Dive into the architecture & building real-world applications leveraging NLP Models starting from RNN to Transformer.

53 min read

deepak kumar

READ MEDIUM

Translate to

With the advent of LLM, RAG has become goto method using which we are able to use with LLM on dataset on which llm are not trained. Dataset...

16 min read

Free OpenAI o1 chat Try OpenAI o1 API