Niyomukiza Thamar

Weekly Challenge: Week 4

Date : 14/09/2022

Interim Submission

Prompt Engineering: In-context learning with GPT-3 and other Large Language Models

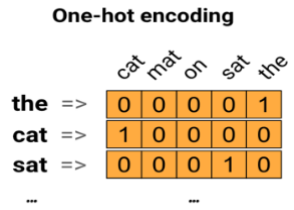
I. Business objective

In-context learning, popularized by the team behind the GPT-3 LLM, brought a new revolution for using Large Language Models in many tasks that the LLM was not originally not trained for. With in-context learning, LLMs are able to constantly adjust their performance on a task depending on the prompt - from structured input that can be considered partly a few-shot training and partly a test input. This has opened up many applications. The aim is to use these pretrained big models to be able to recommend if they can be used to extract relevant entities from job descriptions and to classify web pages given only a few examples of human scores.

II. Understanding Concepts

Before starting to explain about word embedding and its types like word2vec, it is worthy understanding what word embedding is because it is the foundation of text analysis. As we all know, computers understand zeros and ones, but in this case, we want to work on text for us to preform machine learning tasks. Consequently, that push us to think and come up with ways to represent text or strings into numbers. Moreover, machine learning models accept input vectors specifically arrays of numbers to process them into outputs. This can only be done by converting strings into tokens. Then convert tokens into arrays of numbers before feeding them into a model.

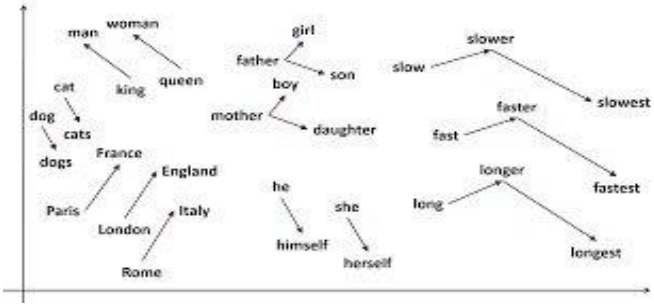
One way is on converting tokens into numbers is using "**one-hot**" to encode each word in your vocabulary to create a vector that contains the encoding of the sentence.



Another way is **word embeddings** where words with similar meaning (e.g., Apple and mango are related because they are all fruits) have similar representation. Individual words are represented as real-valued vectors in a predefined vector space. Word embeddings provides a way to use an efficient, dense representation in which similar words have a similar encoding. They are not computed manually, and weights are learned by the model during training, in the same way a model learns weights for a dense layer. When working on a natural language processing project, we have more than one type of word embeddings techniques that we can use.. In this specific paper, we will be focusing on Word2Vec.

Word2vec is word embedding type of vector space model that takes as input the linguistic context of words in some corpus of text, and outputs an N dimensional space of these words where each word is represented as a vector of dimension N in that space. Word2vec model projects features on a plane, which is very hard to see in the raw data. Words can be projected on the vector space

so that semantic relations of various types of words are easily seen. Moreover, turning texts into a vector space facilitate the mathematic operations, hence machine learning engineer could apply vector arithmetic on these words and achieve useful results. Below is the graphical representation:



Examples of various semantic and grammatical relations that are captured in the vector space.: *https://www.quora.com/Is-word-embedding-word2vec-a-type-of-vector-space-model*

Vector arithmetic examples:

- [king] - [man] + [woman] ~= [queen]
- [walking] - [swimming] + [swam] ~= [walked]

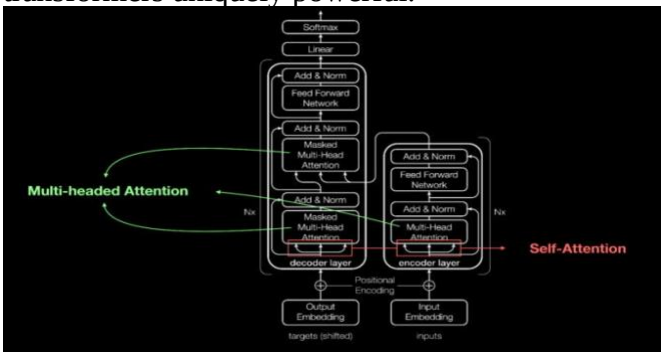
Although word2vec as well as other technologies used in NLP have achieve great things, they also have some cons to consider. According to this paper Man is to Computer Programmer as Woman is to Homemaker? Debiasing Word Embeddings, the presence of gender stereotypes and other biases in word embeddings such as word2vec exist. Unfortunately, it appears that these models inherit and amplify the biases and stereotypes that appear in the original text.

Positional embedding

Normally, the meaning of a sentence is associated with the order of the words and their positions. It can change if someone re-ordered the words in a sentence. Using natural language processing with the recurrent neural networks, the inbuilt mechanism deals with the order of sequences. However, the transformer model does not use RNN or CNN. Every data point is taken as an independent entity. Thus, information about the position of the word is explicitly added to the model to maintain the information regarding the order of words in a sentence. Positional encoding is a way the information of order of objects in a sequence is retained.

Transformer algorithms

Transformers are the deep learning models like recurrent neural networks (RNNs) they are designed for handling the sequential data, such as natural language, for tasks like translation and text summarization. The main aim of transformer is to solve sequence to sequence task while handling long range dependencies with ease. They are basically large encoder/decoder blocks that process data. Small but strategic additions to these blocks (shown in the diagram below) make transformers uniquely powerful.



https://blogs.nvidia.com/blog/2022/03/25/what-is-a-transformer-model/

A decoder is a self-regressor and can't see the future words. On the other hand, an encoder in transformer is a self-regressor, which means it will predict the next token according to the previous. So, the input can't see the future words. we use masked multi-head attention to do this.

Attention and self-attention algorithms

Self-attention is a general way of learning relationships. The attention mechanism was introduced to improve the performance of the encoder-decoder model for machine translation. The idea behind the attention mechanism was to permit the decoder to utilize the most relevant parts of the input sequence in a flexible manner, by a weighted combination of all of the encoded input vectors, with the most relevant vectors being attributed the highest weights.

GPT-3 as a generative language models work.

The third generation Generative Pre-trained Transformer is a neural network machine learning model trained using huge internet data to generate any type of text. It only requires a small amount of input text to generate large volumes of relevant and sophisticated machine-generated text. It has a neural network machine learning model that can take input text as an input and transform it into what it predicts the most useful result will be. This is accomplished by training the system on the vast body of internet text to spot patterns. More specifically, GPT-3 is the third version of a model that is focused on text generation based on being pre-trained on a huge amount of text. When a user provides text input, the system analyzes the language and uses a text predictor to create the most likely output. Even without much additional tuning or training, the model generates high-quality output text that feels similar to what humans would produce. In terms of size GPT-3 is enormous compared to BERT as it is trained on billions of parameters '470' times bigger than the BERT model. BERT requires a fine-tuning process in great detail with large dataset examples to train the algorithm for specific downstream tasks

In-context and fine-tuning

In-context learning is aN emergent behavior in large language models (LMs) where the LM performs a task just by conditioning on input-output examples, without optimizing any parameters. Vanilla methods for training large language models such as GPT-3 and BERT require the model to be pre-trained with unlabeled data and then fine-tuned for specific tasks with labeled data. Fine-tuning is one approach to transfer learning where you change the model output to fit the new task and train only the output model In contrast, prompt-based learning models can autonomously tune themselves for different tasks by transferring domain knowledge introduced through prompts.
Few-shot learning is used in prompt learning we learn to train the dataset with lower or limited information. Zero shot means no examples or structure to follow have been given to the model

In brief, the LLMs are the future and specialized people in prompt engineering demand will grow fast. It is time to start getting familiar with these technologies and use them for the business objectives fulfillment.

Reference:

1. https://www.tensorflow.org/text/guide/word_embeddings

2. https://www.quora.com/Is-word-embedding-word2vec-a-type-of-vector-space-model
3. *https://www.quora.com/Is-word-embedding-word2vec-a-type-of-vector-space-model*
4. https://blogs.nvidia.com/blog/2022/03/25/what-is-a-transformer-model