An Overview of Natural Language Processing

From Word2Vec To GPT

Jaihua Yen

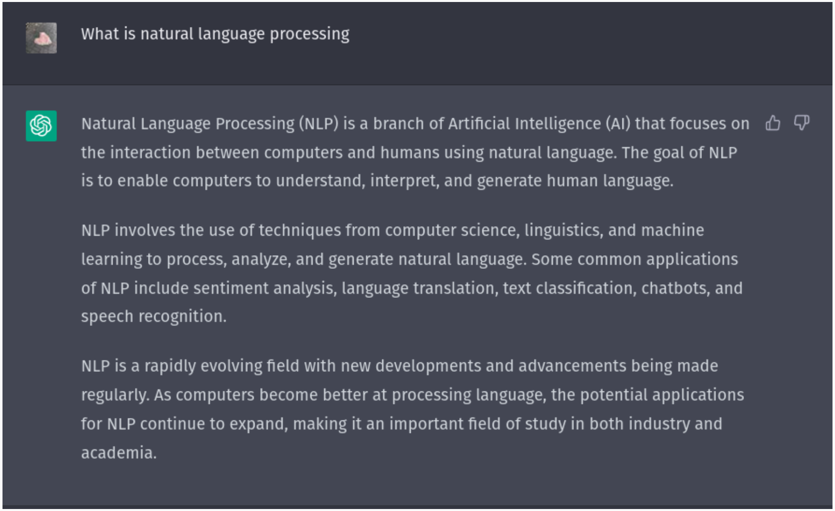
March 14, 2023

QNAP

Table of contents

Introduction

Generative AI



What is natural language processing

Natural Language Processing (NLP) is a branch of Artificial Intelligence (AI) that focuses on the interaction between computers and humans using natural language. The goal of NLP is to enable computers to understand, interpret, and generate human language.

NLP involves the use of techniques from computer science, linguistics, and machine learning to process, analyze, and generate natural language. Some common applications of NLP include sentiment analysis, language translation, text classification, chatbots, and speech recognition.

NLP is a rapidly evolving field with new developments and advancements being made regularly. As computers become better at processing language, the potential applications for NLP continue to expand, making it an important field of study in both industry and academia.

Applications in NLP

- Natural Language Processing (NLP) is now maturely developed in many tasks.
    1. Documents could be classified into several caegories for as to search files easily.
    2. Text could be generatived by AI such as GPT series and ChatGPT.
- In general, NLP is used in the following several scenarios:
    1. Named-Entity Recognition
    2. Document Classification
    3. Machine Translation
    4. ChatBot
    5. Text Semantic Analysis
    6. Text Generation

TF-IDF

Defects of Word Frequency

- The central idea of the NLP is how to quantify the content of the document.
- Term-frequency (TF) could be a measure to the document, but words inside such as ”the” and ”this” will be regard as an important words since they occured in the document many times.
- Another approach is to look at the inverse document frequency (IDF) of the word.
    1. Decreases the weight for commonly occured words
    2. Increases the weight for words that are not commonly occured in documents

TF-IDF

- Now we're going to define the term-frequency and inverse document frequency:

Definition (Term-Frequency)

Term-Frequency is the frequency of the word t in the document d which can count as follows:

$$tf_{t,d} = \frac{n_{t,d}}{\sum_{k \in d} n_{k,d}}$$

where $n_{t,d}$ is the number of words t in the document d.

Definition (Inverse Document Frequency)

Inverse Document Frequency is defined as follows:

$$idf_t = log(\frac{N}{df_t})$$

where N is the number of documents and df_t is the the number of documents where word t occurs.

TF-IDF

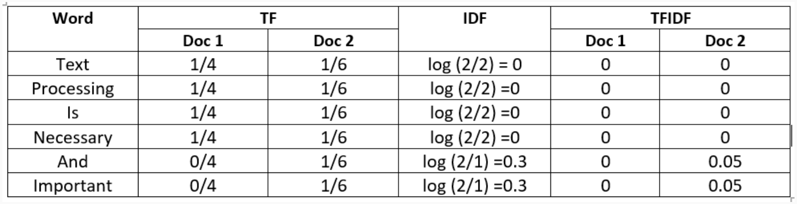
Definition (TF-IDF Score)

The TF-IDF Score of the word t in the document d is defined as follows:

$$tfidf_{t,d} = tf_{t,d} \times idf_t$$

Example

- Here we have an example of implementing tf-idf in these three sentences:
  1. Text processing is necessary.
  2. Text processing is necessary and important.
  3. Text processing is easy.
- The result would be:



Word	TF		IDF	TFIDF	
	Doc 1	Doc 2		Doc 1	Doc 2
Text	1/4	1/6	log (2/2) = 0	0	0
Processing	1/4	1/6	log (2/2) =0	0	0
Is	1/4	1/6	log (2/2) =0	0	0
Necessary	1/4	1/6	log (2/2) =0	0	0
And	0/4	1/6	log (2/1) =0.3	0	0.05
Important	0/4	1/6	log (2/1) =0.3	0	0.05

[Source of Example]

Word2Vec

Issues in Word Representation

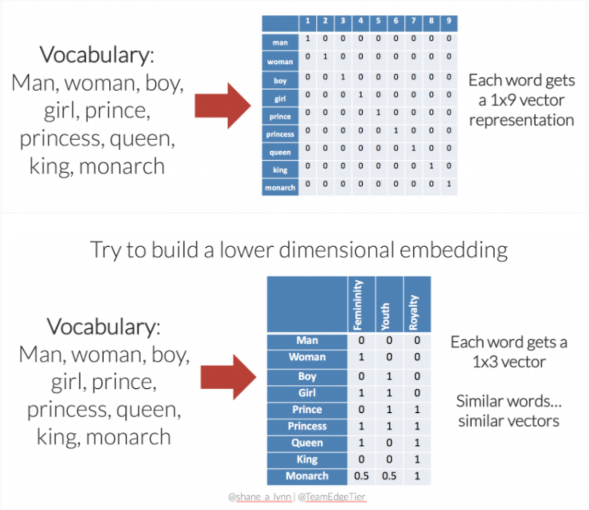
- Words are often represented as one-hot encoding in computer.
- For example, we can set hotel and motel to two different representative as one-hot encoding:

$$v_{motel} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix},\ v_{hotel} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix},$$

Issues in Word Representation

- However, there are several issues when words are represented as one-hot encoding:
    1. We cannot gain information of relation between words.
        - We have no idea how motel and hotel are relate to each other while those two vectors are orthogonal! i.e. $\langle v_{motel},\ v_{hotel} \rangle = 0$
    2. The vector would be very sparse if there exists large amount of unique words.
- That's why we introduce word embedding approach to tackle these problems. [1]

Word Embedding



Vocabulary:
Man, woman, boy, girl, prince, princess, queen, king, monarch

	1	2	3	4	5	6	7	8	9
man	1	0	0	0	0	0	0	0	0
woman	0	1	0	0	0	0	0	0	0
boy	0	0	1	0	0	0	0	0	0
girl	0	0	0	1	0	0	0	0	0
prince	0	0	0	0	1	0	0	0	0
princess	0	0	0	0	0	1	0	0	0
queen	0	0	0	0	0	0	1	0	0
king	0	0	0	0	0	0	0	1	0
monarch	0	0	0	0	0	0	0	0	1

Each word gets a 1x9 vector representation

Try to build a lower dimensional embedding

Vocabulary:
Man, woman, boy, girl, prince, princess, queen, king, monarch

	Femininity	Youth	Royalty
Man	0	0	0
Woman	1	0	0
Boy	0	1	0
Girl	1	1	0
Prince	0	1	1
Princess	1	1	1
Queen	1	0	1
King	0	0	1
Monarch	0.5	0.5	1

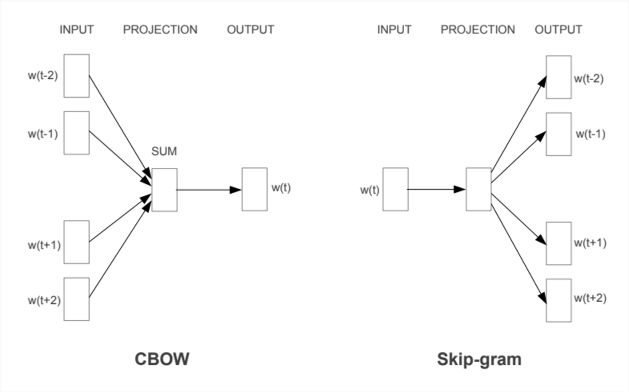
Each word gets a 1x3 vector

Similar words... similar vectors

@shane_a_lynn | @TeamEdgeTier

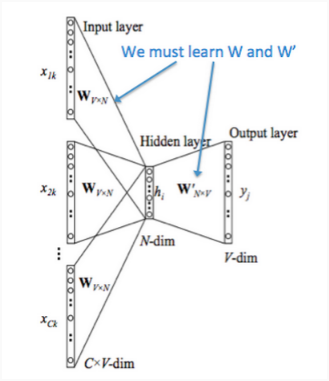
[Image Source]

Word2Vec



[Image Source]

CBOW



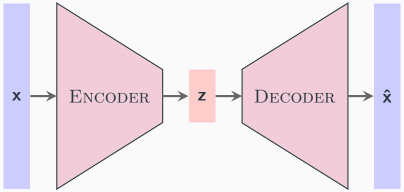
[Image Source]

- Use probability $P(y_i|x_{1k}, x_{2k}, ..., x_{Ck})$ to learn the weight matrix W!
- W is used to be the pre-trained model when we transform the words into embedding vectors in the unseen documents.

Transformer

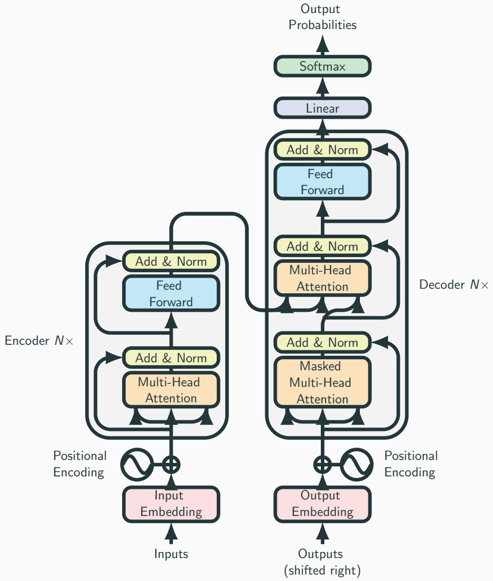
Sequence to Sequence (Seq2Seq)

- Many NLP tasks viewed sequence-to-sequence:
  1. Summarization (whole document $\rightarrow$ shorter text)
  2. Machine Translation (one language $\rightarrow$ target language)



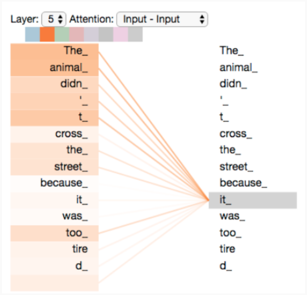
- For example, $\mathbf{x}$ is the input sequence (input je suis étudiant) and $\hat{\mathbf{x}}$ is the output sequence (I am a student).
- Seq2Seq is constructed by encoder and decoder.
  1. Encoder: Decode the meaning of the source text.
  2. Decoder: Re-encode the meaning to the target language.

Architecture



Attention (Self-Attention Mechanism)

- Assume we want to translate a sentence: "The animal didn't cross the street because it was too tired"



[Image Source]

- Advantages:
    1. Interaction distance
    2. Parallelizability
    3. Interpretability

BERT

Defects of Word2Vec

-

GPT

Conclusion

Questions?

References i

T. Mikolov et al.
Efficient estimation of word representations in vector space.
arXiv preprint arXiv, 1301.3781, 2013.