Abstract

Deep learning techniques utilize different processing layers to comprehend hierarchical representations of data. In natural language processing (NLP), a variety of model designs and methods have bloomed in recent times. In this paper, we evaluate important deep learning related models and methods that have been assigned for several NLP assignments and deliver the path of their progression. Comprehensive knowledge of the past, present, and future of deep learning in NLP is proposed. We furthermore correlate, contrast, and shorten the several models.

Index

Natural language processing (NLP)

Natural language processing (NLP) utilizes computer algorithms and artificial intelligence to enable computers to recognize and respond to human communication. It is an attempt to make computers intelligent by making humans believe they are interacting with another human. While several NLP methods exist, they typically involve breaking speech or text into discrete sub-units and then comparing these to a database of how these units fit together based on experience. It was formulated to build software that generates and comprehends natural languages so that a user can have natural conversations with a computer instead of through programming or artificial languages like Java or C.

Examples of NLP: Amazon Echo, Text-to-speech apps, which are now found on most iOS and Android platforms

Stages of Natural Language Processing (NLP)

Stage 1:

The first task of NLP is to understand the natural language received by the computer. A built-in statistical model is used to perform a speech recognition routine that converts the natural language to a programming language.

Stage 2:

The next task is called the part-of-speech (POS). The grammatical form of a word is identified in this process i.e., nouns, verbs, adjectives, tenses, etc. Now the computer can understand the meaning of the speech that was made.

Stage 3:

The final task is text-to-speech conversion. At this stage, the computer programming language is converted to the audible or textual format.

Word2Vec

Word embeddings were revolutionized by Mikolov and his co-workers, it is one of the most popular techniques for Vector representation of a particular word.

Consider the following similar sentences: **Have a good day** and **Have a great day**. They hardly have different meanings. If we construct an exhaustive vocabulary (let's call it V), it will have V = {Have, a, good, great, day}.

Assume a one-hot encoded vector

Have = [1,0,0,0,0] ; a=[0,1,0,0,0] ; good=[0,0,1,0,0] ; great=[0,0,0,1,0] ; day=[0,0,0,0,1] ;

$$V = \begin{matrix} [1,0,0,0,0] \\ [0,1,0,0,0] \\ [0,0,1,0,0] \\ [0,0,0,1,0] \\ [0,0,0,0,1] \end{matrix}$$

If we try to visualize these encodings, we can think of a 5-dimensional space, where each word occupies only one of the dimensions. This means 'good' and 'great' are as different as 'day' and 'have', which is not true.
Our objective is to have words with similar contexts occupy close spatial positions. To overcome this, we use the CBOW (Common Bag Of Words) model

CBOW Model
CBOW model enumerates the conditional probability of a target word given in the context word neighboring it the skip-gram model does the contrasts of the CBOW model, by predicting the surrounding context words

given the central target word. The context words are supposed to be located symmetrically to the target words within a distance one at the same to the window size in both directions. In unsupervised settings, the word embedding dimension is resolved by the accuracy of prediction. As the embedding dimension increases, the accuracy of prediction also increases. The CBOW model is a simple fully connected neural network with one hidden layer.

The input layer- takes the one-hot vector of context word has V neurons

the hidden layer- has N neurons.

The output layer- is the softmax probability of overall words in the vocabulary.

Finally, the three layers are connected by a weight matrix

W ∈ RV ×N and W

0

∈ RH×V