UDC 004.8

Олег Ігорович Онуфрик

Word2vec as the approach in natural language processing

Анотація. За останнє десятиліття штучний інтелект зробив великий крок у розвитку. Сучасна людина щодня зустрічається з ним в побуті. Штучний інтелект складається з багатьох напрямків, таких як – комп'ютерний зір, розпізнавання мови, тексту, тощо. Алгоритми обробки природньої мови займають значну нішу в галузі штучного інтелекту, зокрема такі компанії, як Google, Apple and Amazon використовують їх в своїх голосових помічниках.

Ключові слова. Штучний інтелект, алгоритми, природня мова, аналіз.

Introduction. Natural language processing is a field that combines artificial intelligence, computer science, computational linguistics and explores the problem of analysis and synthesis of natural language. Natural language processing algorithms allow the creations of intelligent systems that are able to generate, translate and determine the emotional color of the text, communicate with the end-user(chat-bot systems), and voice assistants (Alexa, Google Assistant).

Word2vec is a technique for natural language processing. It uses neural network to learn word associations from large datasets. Once the model is trained – it can suggest words for a sentence. Itself, word2vec represents each distinct word with a particular list of numbers called a vector. Word2vec can utilize either of two architectures: continuous bag-of-words or continuous skip-gram.

The intuition of word2vec is that instead of counting how often each word ω occurs near Ω we will instead train a classifier on a binary prediction task. "Is a word ω likely to show up near Ω?" We do not actually care about this prediction task, instead we will take the learned classifier weights as the word embeddings. The benefit of this approach is – we can just use running text as implicitly supervised training data for a classifier. A word ώ that occurs near to the word Ω considered as correct answer to our question "Is a word ω likely to show up near Ω?", so that, need for any sort of labeled data disappear.

Word2vec learns embeddings by starting with an initial set of embedding vectors and then iteratively shifting the embedding of each word ω to be more like the embeddings of words that occur nearby texts, and less like the embeddings of words that do not occur nearby. For training a binary classifier it is require having negative examples. Basically, skip-gram uses more negative examples that positive ones. For each pair (t, c) training instances we will create *k*-negative examples, each containing of the target word *t* and a noise word. Noise word means a random word from the lexicon that is not equal to the target word *t*.

Continuous bag-of-words takes the context of each word as the input and tries to predict next word corresponding to the context. For example – the sentence is "Have a good night" and the input to the neural network is the word "good". Here we are trying to predict a target word using a single input word "good". By predicting the target word – we learn the vector representation of the target word. The input is a vector of size *V*. The hidden layer contains *N* neurons and output layer – a vector of size *V*. The hidden layer neurons copy the weighted sum of inputs to the next layer. The described model uses single context word to predict the target. Also, it is possible to use multiple context words. The only difference is – to calculate hidden layer inputs, model take an average over all the C context word inputs.

Conclusions. During the research was found advantages and disadvantages word2vec approach in natural language processing. Both, continuous bag-of-words, and continuous skip-gram defined by word2vec are good models to predict target word in sentence. Continuous bag-of-words architecture – the model predicts the target word from a surrounding context word. Continuous skip-gram architecture – the model uses the target word to predict the surrounding context words. Continuous bag-of-words architecture is faster that continuous skip-gram architecture, but continuous skip-gram architecture is better for infrequent words.

References

1. Jurafsky Daniel. Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. Daniel Jurafsky, James H. Martin. - 3rd edition. - Prentice Hall, 2019. - 621 p.
2. Natural Language Processing with Python – Analyzing Text with the Natural Language Toolkit. Steven Bird, Ewan Klein, and Edward Loper