



Sentiment Analysis Using Word2Vec, FastText and Universal Sentence Encoder in Keras

Jatin Mandav

[Follow](#)

Data Science Intern at Shar...

👍 18

💬 1

🔄 0

Opinion mining (sometimes known as sentiment analysis or emotion AI) refers to the use of natural language processing, text analysis, computational linguistics, and biometrics to systematically identify, extract, quantify, and study affective states and subjective information. Sentiment analysis is widely applied to voice of the customer materials such as reviews and survey responses, online and social media, and healthcare materials for applications that range from marketing to customer service to clinical medicine.[Source: [Wikipedia](#)]

Sentiment analysis is performed on **Twitter Data** using various word-embedding models namely: Word2Vec, FastText, Universal Sentence Encoder.

Requirements: TensorFlow Hub, TensorFlow, Keras, Gensim, NLTK, NumPy, tqdm

The analysis is performed on **400,000 Tweets** on a **CNN-LSTM** DeepNet.

The Entire Project is available at GitHub: [Neural-Networks - All about Neural Networks/github.com](#)

Architecture Model (Generated by TensorBoard):

In my experience with all the three models, I observed that word2vec takes a lot more time to generate Vectors from all the three models. FastText and Universal Sentence Encoder take relatively same time. For word2vec and fastText, pre-processing of data is required which takes some amount of time. When it comes to training, fastText takes a lot less time than Universal Sentence Encoder and as same time as word2vec model.

But as you can see, the accuracy by Universal Sentence Encoder is much more higher than any of the two models.

Universal Sentence Encoding sound very promising :) but on the drawbacks, it takes a significant amount of time to train or even complete one epoch.

Let's dig into each model!

Word2Vec

Accuracy Achieved: **Approx 69%**

Python Notebook: [Neural-Networks - All about Neural Networks/github.com](#)

Word2vec is a group of related models that are used to produce word embeddings. These models are shallow, two-layer neural networks that are trained to reconstruct linguistic contexts of words. Word2vec takes as its input a large corpus of text and produces a vector space, typically of several hundred dimensions, with each unique word in the corpus being assigned a corresponding vector in the space. Word vectors are positioned in the vector space such that words that share common contexts in the corpus are located in close proximity to one another in the space. [Source: [Wikipedia](#)]

Docs in Gensim: [models.word2vec](#)

Word2Vec has 2 important models inside: Skip-Grams and Continous Bag-of-Words(CBOW)

Skip-Grams:

In Skip-Gram model, we take a centre word and a window of context words or neighbors within the context window and we try to predict context words for each centre word. The model generates a probability distribution i.e., probability of a word appearing in context given centre word and the task here is to choose the vector representation to maximize the probability.

Continous Bag-of-Words (CBOW):

CBOW is opposite of Skip-Grams. We attempt to predict the centre word from the given context i.e., we try to predict the centre word by summing vectors of surrounding words.

fastText

Accuracy Achieved: **Approx 69%**

Python Notebook: [Neural-Networks - All about Neural Networks/github.com](#)

fastText is a library for learning of word embeddings and text classification created by Facebook's AI Research (FAIR) lab. The model is an unsupervised learning algorithm for obtaining vector representations for words. Facebook makes available pretrained models for 294 languages. fastText uses Neural network for word embedding.

Docs on Gensim: [models.fastText](#)

FastText is an extension to Word2Vec proposed by Facebook in 2016. Instead of feeding individual words into the Neural Network, FastText breaks words into several n-grams (sub-words). For instance, the tri-grams for the word apple is app, ppl, and ple (ignoring the starting and ending of boundaries of words). The word embedding vector for apple will be the sum of all these n-grams. After training the Neural Network, we will have word embeddings for all the n-grams given the training dataset. Rare words can now be properly represented since it is highly likely that some of their n-grams also appears in other words. I will show you how to use FastText with Gensim in the following section.

Universal Sentence Encoder

Accuracy Achieved: **Approx 77%**

Python Notebook: [Neural-Networks - All about Neural Networks/github.com](#)

Released in 2018, The Universal Sentence Encoder encodes text into high dimensional vectors that can be used for text classification, semantic similarity, clustering and other natural language tasks.

The model is trained and optimized for greater-than-word length text, such as sentences, phrases or short paragraphs. It is trained on a variety of data sources and a variety of tasks with the aim of dynamically accommodating a wide variety of natural language understanding tasks. The input is variable length English text and the output is a **512 dimensional vector**. We apply this model to the STS benchmark for semantic similarity, and the results can be seen in the example notebook made available. The universal-sentence-encoder model is trained with a deep averaging network (DAN) encoder.

Further Readings:

- [Universal Sentence Encoder](#)
- [The Current Best of Universal Word Embeddings and Sentence Embeddings](#)
- [Vector Representations of Words](#)
- [Advances in Semantic Textual Similarity](#)

[fastText](#)

[Word2Vec and FastText Word Embedding with Gensim](#)

Publicado por

Jatin Mandav

Data Science Intern at ShareChat

Seguir

V

#SentimentAnalysis #MachineLearning #DeepLearning #word2vec #fastText #UniversalSentenceEncoder #Twitter #Keras #TensorFlow

1 comment

[Sign in](#) to leave your comment

Surya Vamsi Tenneti

SDE Intern @ Amazon | Masters in Computer Science @ ASU

2 años

Nicell!

Recomendar

Responder

1 recomendación

Más de Jatin Mandav 1 artículo

© 2020

Acerca de

Condiciones de uso

Política de privacidad

Política de cookies

Política de copyright

Política de marca

Controles de invitados

Directrices comunitarias

Idioma