Sentiment Analysis Using Deep Learning Technology

WANG, JIN

This presentation will give an overview of sentiment analysis and review the latest technology in this field. It will outline how to use the deep learning framework TensorFlow and Keras to perform sentiment analysis and apply them to use the service desk sample data.

MAY 10, 2019

Agenda

- Introduction
- How Does Sentiment Analysis Work?
- Deep Learning for Sentiment Analysis
- Step by Step Data Analysis using Python
- Summary

What is Sentiment Analysis?

The automated process of understanding an opinion about a given subject from written or spoken language. It tries to identify and extract opinions within text and following attributes of the expression.

- *Polarity*: if the speaker express a *positive* or *negative* opinion,
- *Subject*: the thing that is being talked about,
- *Opinion holder*: the person, or entity that expresses the opinion.

Sentiment Analysis is a Classification Problem

Two sub-problems to be resolved:

- Classifying a sentence as *subjective* or *objective*, known as **subjectivity classification**.
- Classifying a sentence as expressing a *positive*, *negative* or *neutral* opinion, known as **polarity classification**.

Sentiment Analysis Scope

- **Sub-sentence level** sentiment analysis obtains the sentiment of sub-expressions within a sentence.
- **Sentence level** sentiment analysis obtains the sentiment of a single sentence.
- **Document level** sentiment analysis obtains the sentiment of a complete document or paragraph.

Sentiment Analysis Use Cases

- Social media monitoring
- Brand monitoring
- Voice of customer (VoC)
- Customer service
- Workforce analytics and voice of employee
- Product analytics
- Market research and analysis

Agenda

- Introduction
- How Does Sentiment Analysis Work?
- Deep Learning for Sentiment Analysis
- Step by Step Data Analysis using Python
- Summary

Sentiment Analysis Approaches

- **Rule-based** systems that perform sentiment analysis based on a set of manually crafted rules.
- **Automatic** systems that rely on machine learning techniques to learn from data.
- **Hybrid** systems that combine both rule based and automatic approaches.

Sentiment Analysis Algorithms

- **Naïve Bayes**: A family of probabilistic algorithms that uses Bayes's Theorem to predict the category of a text.
- **Linear Regression**: A very well-known algorithm in statistics used to predict some value (Y) given a set of features (X).
- **Support Vector Machine**: A non-probabilistic model which uses a representation of text examples as points in a multidimensional space.
- **Deeping Learning**: A diverse set of algorithms that attempts to imitate how the human brain works by employing artificial neural networks to process data.

Sentiment Analysis APIs for Python

- **NLTK** – a traditional NLP library
- **Spacy** - another recent NLP library
- **Scikit-learn** – a library for Machine Learning
- **TensorFlow** – a big framework that performs deep learning
- **Keras** - provides abstractions to work with multiple neural networks. It can be run on top of Tensorflow and Theano
- **PyTorch** - a recent Deep Learning framework

Sentiment Analysis APIs for Java

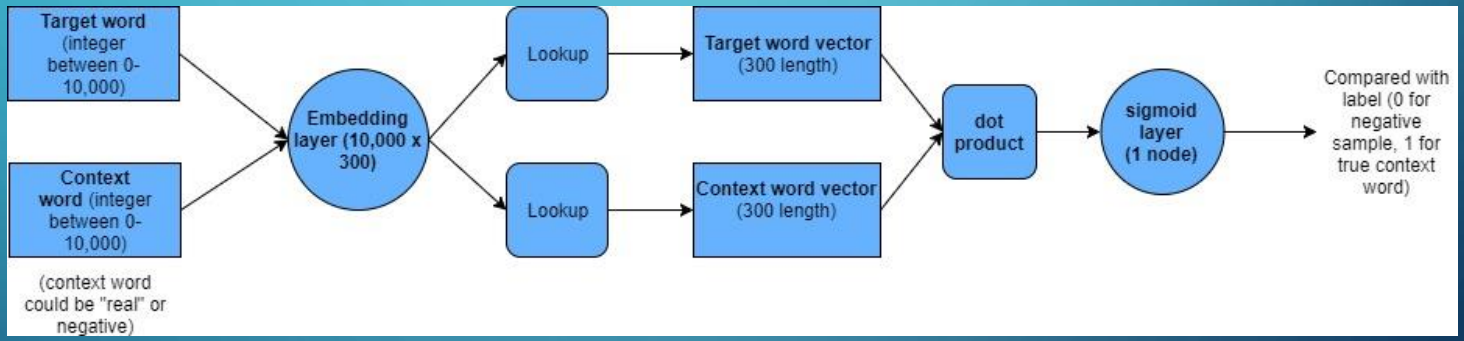
- **OpenNLP** – a toolkit for NLP tasks
- **Stanford CoreNLP** - a Java suite of core NLP tools
- **Lingpipe** – a Java toolkit for processing text using computational linguistics.
- **Weka** – a set of tools created by The University of Waikato for NLP tasks.

Agenda

- Introduction
- How Does Sentiment Analysis Work?
- Deep Learning for Sentiment Analysis
- Smart Features for the Service Desk
- Summary

Word Vector Representation

- Word meaning is defined in terms of vector
- Distributional similarity based representations



Word Vector Algorithms

- Predict between every word and its context words!
- Two algorithms

  1. **Skip-grams (SG)**

     Predict context words given target (position independent)

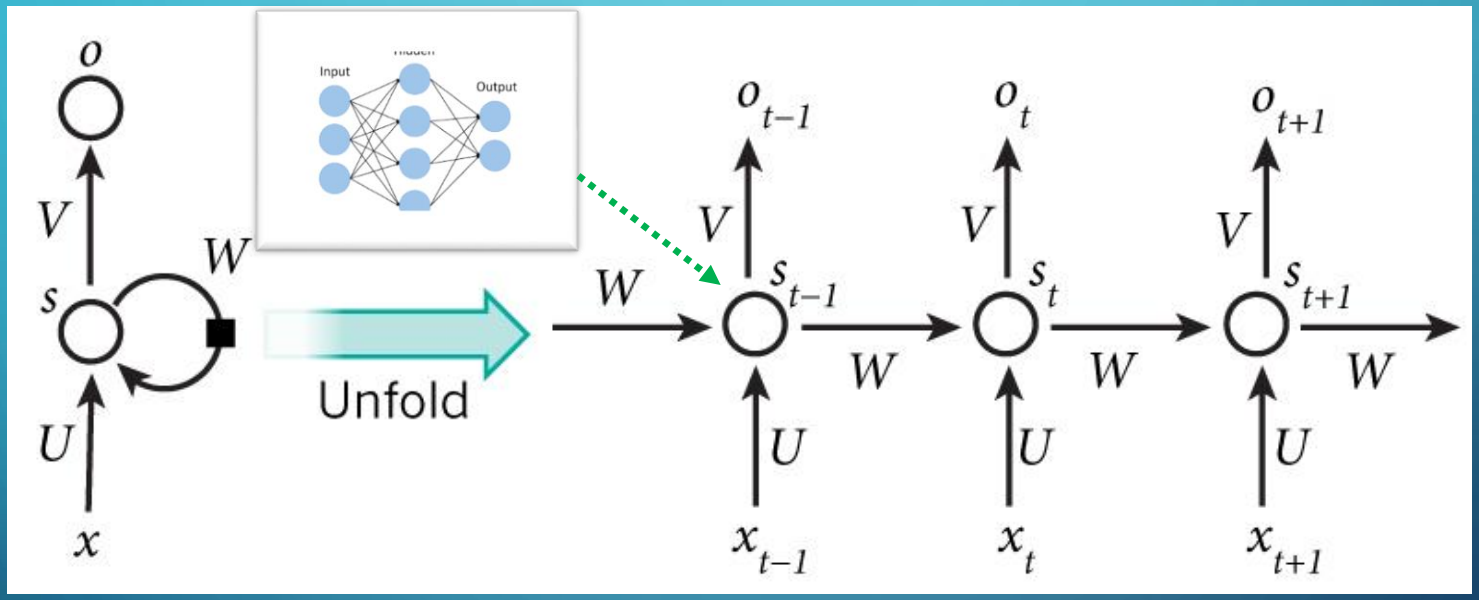
  2. **Continuous Bag of Words (CBOW)**

     Predict target word from bag-of-words context

- Eg: "the cat ***sat*** on the mat"

Recurrent Neural Networks (RNNs)

The idea behind RNNs is to make use of sequential information.

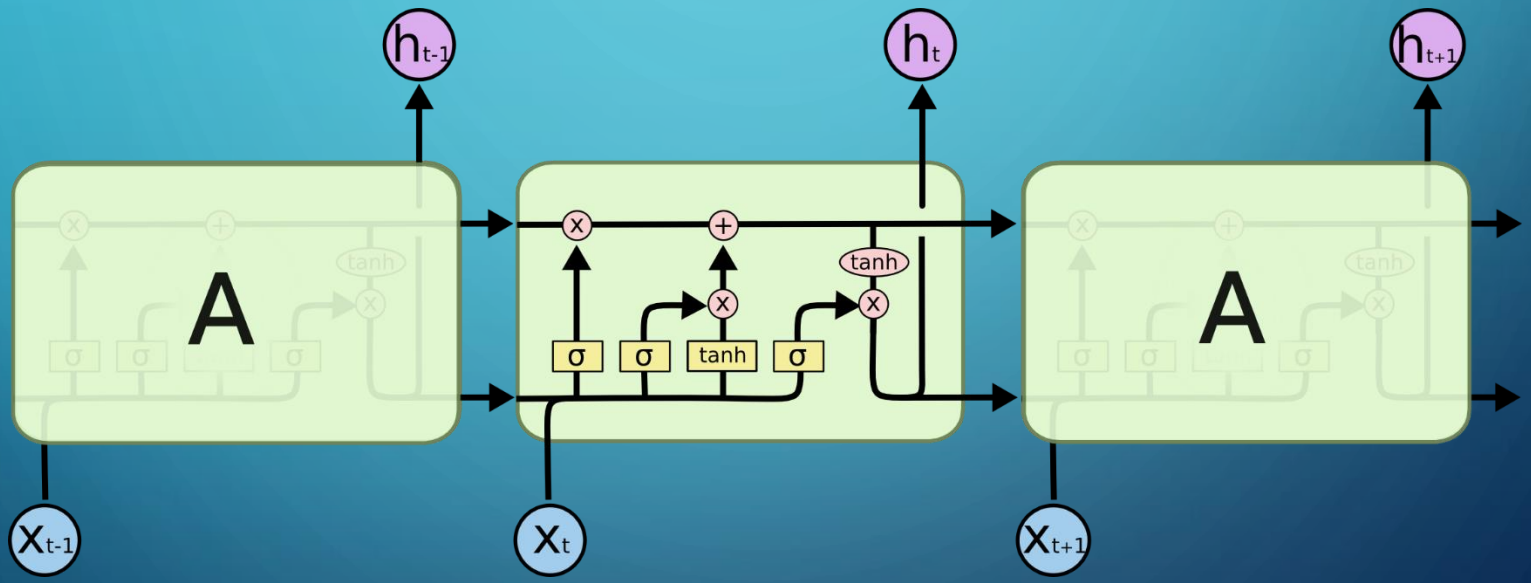


In theory, RNNs are absolutely capable of handling “long-term dependencies.”

In practice, RNNs don’t seem to be able to learn them.

Long Short Term Memory (LSTM) Networks

- A special kind of RNN - The key to LSTMs is the cell state, the horizontal line running through the top of the diagram.



Agenda

- Introduction
- How Does Sentiment Analysis Work?
- Deep Learning for Sentiment Analysis
- Step by Step Data Analysis using Python
- Summary

Framing Sentiment Analysis as a Deep Learning Problem

1. Loading pretrained word vectors (optional)
2. Preprocessing documents
3. Create RNN model with LSTM units
4. Training sentiment model with labelled documents
5. Testing sentiment model with a separate labelled documents
6. Predicting sentiment for IT Service Desk tickets (or other docs)

Preprocessing Documents

- Load public available data sets called “IMDB movie review set”, which can be downloaded at http://ai.stanford.edu/~amaas/data/sentiment/
- Tokenize all documents – Splitting docs to words:
  - *maxNumWords* – only use top N words
- Encode all documents – converting words to index:
  - *maxSeqLength* - Maximum number of words for each document

Create RNN Model with LSTM Units

- Load preprocessed data – Word2Index dictionary and WordID Matrix for all documents
- Build LSTM RNNs using Keras deep learning library

numDimensions – dimensions for each word vector

lstmUnits – number of neurons in hidden layer

numClasses – classes to be classified

```
model = Sequential()
model.add(Embedding(maxNumWords, numDimensions, input_length=maxSeqLength))
model.add(LSTM(lstmUnits, recurrent_dropout=0.25))
model.add(Dense(2, activation='softmax'))
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
```

Train Sentiment Model with Labelled Documents

- **Enable tensorboard**

  tensorboard = TensorBoard(log_dir="logs/{}".format(time()))

- **Define checkpoint**

  filepath="weights-improvement-{epoch:02d}-{val_acc:.2f}.hdf5"

  checkpoint = ModelCheckpoint(filepath, monitor='val_acc', verbose=1, save_best_only=True, mode='max')

- **Define callback list**

  callbacks_list = [checkpoint, tensorboard]

- **Fit the Model**

  model.fit(X_train, y_train, validation_data=(X_test, y_test), epochs=100, batch_size=128, callbacks=callbacks_list, verbose=2)

Test Sentiment Model with Labelled Documents

- Load the best model from checkpoint

  model = load_model('weights-improvement-86-0.83.hdf5')

- Apply loaded model to test data

  loss, acc = model.evaluate(X_test, y_test, verbose=0)

Predicting Sentiment for IT Service Desk Tickets

- Load preprocessed word dictionary
- Load data file to be classified
  - Use text in ITSM incident tickets
- Encode Docs - Converting words to ids based on pre-trained word dictionary
- Load pretrained model
- Predict sentiment scores for input text

Agenda

- Introduction
- How Does Sentiment Analysis Work?
- Deep Learning for Sentiment Analysis
- Step by Step Data Analysis using Python
- Summary

Summary

- Sentiment analysis is an important application in customer support
- Latest deep learning and NLP technologies let sentiment analysis easier and more reliable
- Implement sentiment analysis in service desk or virtual agent will be a differentiator

References

- https://monkeylearn.com/sentiment-analysis/
- https://machinelearningmastery.com/predict-sentiment-movie-reviews-using-deep-learning/
- https://adventuresinmachinelearning.com/word2vec-keras-tutorial/
- http://colah.github.io/posts/2015-08-Understanding-LSTMs/