Technology Review – Entity Sentiment Analysis with Feature Extraction and Word Embeddings

Christine Frandsen – CS410 Fall 2020

Sentiment analysis typically focuses on determining the sentiment of a piece of text as a whole – whether that's a tweet or a news article. However, there are many instances where breaking down sentiment analysis to the entity level can provide useful insight. This may be beneficial when doing sentiment analysis on a political news article where two presidential candidates are portrayed in different manners. Another area this could be useful is determining how a sports team is portrayed in an article. If a team is discussed in a positive light this could be useful for recommendations to users that are fans of that team. If it's negative, perhaps fans wouldn't want to read negative content about their team. However, bringing in entities to sentiment analysis complicates the process. One suggested approach to bringing in entity level sentiment analysis is to use word embeddings to connect a given entity with adjectives, adverbs, and verbs that describe it and derive the associated sentiment. Throughout this review, we will go over the approach of using word embeddings.

Traditionally, there are two primary ways in which sentiment is typically conducted. The first is using a knowledge-based approach. To use the knowledge-based approach, a corpus of words that are labeled as positive and negative are required. Words could also be labeled as objective as well. One common corpus that is publicly available is SentiWordNet, which is an extension of WordNet. We can then compare the words from the input text with the corpus to determine polarity.

The other method traditionally used for sentiment analysis is a language-based approach. This typically takes form of using some method of supervised machine learning. There are various algorithms that are used, but some of the more popular ones are Naïve Bayes and Support Vector Machines (SVM). These algorithms can be trained on some training data and then applied to the text we want to find the sentiment for. Different algorithms have varying performance, Naïve Bayes has been found to consistently have the best results.

In the early 2010s the idea of using word embeddings for sentiment analysis began to become more popular. By using embeddings, the semantics and context of words were better captured with various machine learning approaches and Neural Networks. One of the most popular way to create word embeddings is using the Word2Vec algorithm from genism. There are two primary training algorithms, either continuous bag-of-words (CBOW) or skip-gram. Typically skip-gram is the better choice when dealing with large datasets.

In order to properly handle multiple entities, text that contains more than one entity requires additional processing. It is important to break down the part-of-speech tags for the words in the text. This helps us determine what the entities are. Further, we will focus on using any words that are adverbs, adjectives, or verbs for entity-level feature extraction. Studies have shown that those three parts of speech are the primary indicators of sentiment in text.

Outside of part-of-speech tagging, there's more preprocessing that is required for entity-level sentiment analysis. This includes various things such as changing the case of the words, stemming, and lemmatizing words for more uniformity and consistency.

When using one of the two more traditional methods, stop words would be removed. However, when using the Word2Vec model, stop words are left to allow the algorithm to gain more context. The Word2Vec model is then trained on an unlabeled dataset that is representative of what the expected input text would be.

The general flow of extracting the sentiment from the input text would vary depending on the number of entities. Regardless of number of entities, the first step would be to pre-process the text and conduct part of speech tagging. For the single-entity texts, it's a fairly straightforward process where the text is tokenized and the Word2Vec model is trained. From there we can extract the sentiment from the text. In the case of a multi-entity text, we extract the neighboring words for the entity. This is typically one or two words on either side of the entity. These words are then assigned a polarity score using SentiWordNet.

Once we have assigned sentiment to our input texts we can analyze the performance. A simple way would be to look at accuracy – the total number of texts classified over the total number of texts classified. This does not handle lopsided data well and incentivizes the model to make predictions geared towards one result. Another option would be to use precision or recall. Both of these work on emphasizing the reduction of either false negatives or false positives. This metric can also be combined into an F1 score to get a bit of a more balanced result.

Based on the research done by Sweeney and Padmanabhan using a knowledge-based approach combined with part-of-speech tagging, precision, recall, and F1-score were all 0.68. When classifying using a more hybrid approach by combining word2vec, part-of-speech tagging, and a knowledge-based approach, all of those numbers increased. This indicates that bringing in a more hybrid approach to sentiment analysis helps improve the effectiveness of the model.

As machine learning and sentiment analysis continues to evolve, it is becoming more and more evident that using a hybrid approach allows for improved results. This helps, particularly in the area of multi-entity sentiment analysis as this requires a unique approach. Being able to break things down to individual entities allows for more insight into the text and allows for better understanding of how different entities are portrayed rather than how a piece of text reads as a whole.

References

Sweeney, C., & Padmanabhan, D. (2017). Multi-entity sentiment analysis using entity-level feature extraction and word embeddings approach. *ACL Anthology, Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, 733-740. doi:10.26615/978-954-452-049-6_094