CS229 Final Project Writeup

Localized Explicit Semantic Analysis for Concept-Based Information Retrieval

Francis Lewis
glewis17@cs.stanford.edu

My goal is to, given a word, text fragment, or longer document, output a set of conceptual ideas that best embody the meaning of the text as we humans understand it. The utility of such a method is providing a reliable way to incorporate knowledge of the world into semantic analysis of text; in doing so, we can generate a set of "concepts" that relate to a text fragment. If these same "concepts" could also be mapped to other forms of input, such as images or sound, we would have a reliable and algorithmic approaches to linking related forms of media together.

Previous work has been done in learning an association between words and high-level concepts by estimating probabilities with text frequency occurring in online encyclopedias; however, previous work has relied on either heavy utilization of natural language processing techniques or processing entire corpora of articles. My proposed method instead involves a localized approach, having the algorithm estimate probabilities by trying to find and learn from articles relevant to the query at hand. To test the efficacy of this approach, I train my model on a selection of Twitter data relating to two different hashtags, and measure how well I can distinguish one set of training tweets from the other.

The data used to train the model involves parsing data from the WikiMedia API, which can be downloaded at http://www.mediawiki.org/wiki/API:Main_page. The data relevant to my model is the title of a Wikipedia article, the related plain text word tokens, and the links to other Wikipedia articles. Before learning, I first parse the raw WikiText by removing special tokens such as curly braces and brackets, normalize all words to lower-case, and stop and stem each token. Each Wikipedia article represents some "concept" and has some associated words; by letting the model read lots of articles and determine which words are most relevant to which articles (or which words aren't relevant at all), we can train a model to classify textual fragments based on the strength of the text fragment's correlation to a constellation of Wikipedia articles.

For testing purposes, I used 100 "tweets" from Twitter (https://twitter.com/). I chose this source for my testing data since "tweets" are

assigned natural categories based on a number of special tokens (“hashtags”); because tweets can be modeled as a “livestream” of text data about a particular current event, using “tweets” provides a good opportunity to test the discerning power of localized ESA against events and text too recent for the algorithm to know anything about. My data was divided into two categories of 50 tweets each, one for “#Raiders” and one for “#BerkeleyProtests”. Since these two categories were trending around the same time in approximately a similar geographic location, they collectively provide a good challenge. I trained my model with the first 30 tweets in each category, and then tested with the final 20 in each category.

My number of features for learning concepts-to-words is high, since each Wikipedia article provides new textual data; individual word tokens, along with a weight that represents how frequently a given word appears in an article, comprise the features for learning Wikipedia article titles, while the Wikipedia article titles themselves comprise the raw input features for the Twitter training. I experimented with feature selection in the realm of selecting the vector length of Wikipedia article titles that would give the best results on the Twitter dataset; though my current success has been mixed, it seems as of now that a length of 5 is optimal.

I used primarily generative models in my project. To fit words to concepts, I built an index of words and what articles those words belonged to. I then learned the probability of an article given a particular word by finding the probability of a word given an article weighted by the probability of a word appearing across all articles; in this way, I was able to estimate a probability model for what articles should be associated with what words. I modified this technique by introducing a control parameter for increasing the number of articles I scan for per word; in particular, the control parameter allows for a particular “depth” to be searched, increasing the complexity of the model by allowing me to learn probabilities for words given several articles. The current method I’m using to allow the learning of significance is TFIDF. For classifying Twitter data, I fit a basic Naïve Bayes model to the concept vectors predicted by my ESA algorithm.

In my results table, I have testing error listed along the y-axis and the different parameters I used to train my two models along the x-axis. The "level deep" represents the number of layers of articles I delved per example before learning from them; the "#-vec" represents the feature selection I used in determining what size concept vector would give the best results in training and testing Twitter data.

	1 level deep, 1-vec	1 level deep, 5-vec	1 level deep, 10-vec	2 levels deep, 1-vec	2 levels deep, 5-vec	2 levels deep, 10-vec
Berkeley Test Er.	95%	85%	85%	90%	90%	85%
Raiders Test Er.	100%	95%	100%	100%	100%	100%

I unfortunately have a very large amount of error. This error reaches a minimum when I analyze only one level of articles and use a vector of size 5 to classify Twitter data. I originally hypothesized that I could minimize the error by simply increasing the amount of text data I analyze as well as using larger vectors with more concepts to reflect this increased learning. Interestingly enough, it didn't seem to have any more of an effect than using less article data and smaller vectors. My suspicion is that since the most relevant information is reflected "closest" to the data (i.e. in the first level of article analysis and the first few concepts in a concept vector), then adding both more article data and a larger concept vector only introduces noise that distorts the prediction process. I also noticed that localized ESA almost completely failed on the Raiders test set; I believe it's because the Raider's set contains many more proper nouns than the Berkeley set, for which Wikipedia may not have data and so I wouldn't have been able to classify.

For the future, my first priority is to find a way to be able to process this data quickly enough to allow even more experimenting and testing. I also would like to experiment with using different probabilistic models for text frequency analysis besides TFIDF; in particular, I would like to replace the "IDF" part with a Gaussian distribution by modeling the IDF values for all words across all selected articles as a Gaussian. I would also like to obtain and test on more data from a variety of sources, as I believe localized ESA has the potential to work much better on certain sets than others (as witnessed in the data sets above). Finally, Wikipedia holds a wealth of other information such as edit places, edit times, users, etc; I think finding novel ways to incorporate this data will make localized ESA a very useful semantic analysis tool in the future.

---

http://www.aaai.org/Papers/IJCAI/2007/IJCAI07-259.pdf
http://snowball.tartarus.org/
http://www.nltk.org/
https://twitter.com/?lang=en