June 7, 2016

EXPLORING IMPLICIT APPROACHES TO POS LEARNING

Matt George

Overview

The progression of the digital era has witnessed the emergence of Big Data, and the manifold increase in the information available for the discretion of learning systems. Driven by the increasing prominence of connected devices and social media, the size of the "digital universe" continues to double biannually, and is expected to reach a size of nearly 44 ZB by 2020.[1] However, utilizing this data has brought to the forefront the major issues of data processing—as Moore's law has begun to waver[2]--as well as the creation of correspondingly large output sets from which learners can train upon. While the development of massively parallel and GPU-based computational methods appears to offer some hope in offsetting the losses in CPU gains, it is the latter of these two issues which has rendered a vast amount of data inaccessible.

In the field of Natural Language Processing, Part-of-Speech (POS) learners have been met with a wide variety of success, however, these methods have often been exclusive to proper literary works, with output sets manually and expensively created by hand. It is in choosing such a relatively well-studied topic that I have set out to determine the validity of an approach that utilizes higher variance data—namely from Twitter—and an implicit approach for programmatic output generation. By implicit, I refer to the inherently relational nature of data that is often taken for granted, and it is from the relational structure of certain types of data—here, conversational—that betrays certain assumptions which can be used in the derivation of an output space.

Conversational data—through social media—is quickly emerging as one of the largest areas for information growth, but importantly, it is the simple idea—that individuals talk about the same thing—that provides a powerful means of aiding in the parsing of difficult instances. It is a reasonable assumption that the set of words that appear in a conversation will be used in the same sense of a word—even across different actors, and therefore, the problem of correctly parsing a host of different instances containing a word in a given conversation can reduce to that of simplest instance in the conversation. In this way, conversations can be used so as to resolve lexical ambiguity.

Here I propose the use of conversational data as a means by which to create an input-output space, and by mining Twitter, create a set of trainable data that uses simplistic and exclusionary parsing rules, in an attempt to classify words appearing in difficult-to-parse contexts using kNN.

Past Work

This idea is not a new one, and indeed, the concept of a single sense per discourse has long been explored. A paper published by Bell Labs [Gale, Church and Yarowsky, 1992] found the existence of "a very strong discourse effect", where "if a polysemous word ... appears two or more times in a well-written discourse, it is extremely likely that they will all share the same sense.[3] Later work by Yarowsky has similarly revealed a word to have "only one sense in a given collocation with a probability of 90-99%."[4] While conversational data does differ to some extent from the literary discourses which earlier efforts have focused upon, I argue that the same principles very much apply.

Sample Collection

Data collection occurred over the course of two weeks, utilizing the Twitter REST API to query for Tweets that contained ambiguous words. This list of ambiguous words was created from a word list of 41,000 commonly occurring words—provided by SCOWL[5]--that was pruned against a part-of-speech dictionary in order to find those words which could take multiple different parts-of-speech. The scope of this project has been limited to only those word which are ambiguous among the possibilities of {N, V, M} where N denotes a noun, V a verb, and N a modifier.

For each of these terms, a search was then conducted so as to ascertain the most replied-to ten tweets containing these terms on 1-day intervals five days prior to program execution. For each tweet, the author's profile was then searched against in order to find replies that also contained the search term. Exact numbers were established due to rate-limit constraints.

Each conversation was then parsed into a number of trainable instances, and the bit vector for the selected ambiguous word contained in the instance that, when parsed using simplistic exclusionary methods, yielded the lowest entropy, was then selected to serve as the output for each instance derived from the conversation. This process yielded a total of 906,803 samples. In terms of the attribute and sentence representation for these instances, a bag-of-words style approach has been utilized, with each instance processed so as to feature a list of ids that corresponded to the tokens occurring within each sample. Each instance also kept track of which word was immediately before and after the search term's instantiation. For validation purposes, approximately 10% of this data—randomly selected—was set aside.

Parsing rules

Let W denote an instance's ambiguous word, N denote a noun, V a verb, M a modifier, A an article, and P a nominative pronoun. The following simple exclusionary grammatical rules have been applied in order to determine the instance with the lowest entropy.

A W => Not V

W A => Not M, N

W P => Not M, V

W V => Not M

M W V => Not M, V

V W N => Not N

Algorithm

A k-nearest-neighbor approach was settled upon for use in the experiment, the decision resting upon the relative ease in implementing kNN, its extensibility, as well as the specialized nature of the collected data. Given the experiment's constraints, more computationally expensive learners would have required too much training time.

```
(defun knn (instances proc dist last heap max-n dist-fn)
  (cond ((null instances) return heap))
        ((< (size heap) max-n)
         recur on (heap-push heap last))
        ((< (min heap) dist)
         pop heap, recur on (heap-push heap last))
        (t recur on heap)))
```

Figure 1. Algorithm pseudocode using recursion and a minimum heap—see provided code for the actual implementation. Distance is calculated using a function passed to the method.

The algorithm itself used recursion, iterating through a list instances and pushing up to k neighbors (max-n) onto a minimum heap, popping the smallest value from the heap given a larger-weighted instance to take its place. Distance was calculated by a function passed to the method, with various different distance functions tested for accuracy.

The following were included among the distance measures tested:

A. d-tokens—distance measured as the sum of similar tokens
B. d-prior—distance determined solely by the word previous to the term's instantiation.
C. d-prior-post—distance determined by the word before and after the term's instantiation.
D. d-weighted-tokens—distance measured as the sum of similar tokens, with a weight placed on the tokens immediately before and after the term's instantiation.

Results

Using the data set aside for validation purposes, filtered so as to only include those instances where the parsing rules were able to successfully eliminate all but one potential part-of-speech for a given term, a validation set of approximately 50,000 instances was created. The kNN algorithm was then run with each distance function, the results of showed meager results overall. As expected, the distance functions which utilized more information—i.e. all of the tokens—performed better on the validation set than did methods only making use of the token immediately prior or after.

frequency, overall (ZeroR)	32870	52066	63%
frequency, term	36995	52066	71%
2nn, prior	28721	52066	55%
2nn, prior-post	31058	52066	60%
2nn, tokens	31975	52066	61%
2nn, tokens, weighted	32230	52066	62%
10nn, prior	34323	52066	66%
10nn, prior-post	35483	52066	68%
10nn, tokens	36354	52066	70%
10nn, tokens, weighted	36431	52066	70%
20nn, prior	35797	52066	69%
20nn, prior-post	36369	52066	70%
20nn, tokens	36873	52066	71%
20nn, tokens, weighted	37078	52066	71%

Figure 2. kNN results using different distance measures.

However, while every method was able to easily outperform a simple ZeroR classifier that defaulted to the most frequent case given all instances, each method struggled to do better than a similar measure that instead calculated the per-term frequency.

Conclusion

While past experiments have shown the idea of one sense per discourse to have merit, at least in this case, the results were underwhelming. Even in the best case, 20nn weighted tokens only outperformed term frequency by 0.16%. Indeed, it would appear that the nature of the dataset, as well as the interplay of the data with the included parsing rules, proved problematic.

Ultimately the dataset skewed heavily in favor of certain terms, and per-term data varied significantly, even between 0-2000 instances for the least and most well-endowed terms respectively. In retrospect, this was an almost certain development, and it makes sense that there would be comparatively few instances in which two individuals both use a term such as “absorbent” or “aromatic” in reply to one another. A larger data collection effort that sought to balance the available data for different terms would likely prove more successful in this regard.

Similarly, the parses for terms as a whole also skewed quite dramatically in one direction. The source of this issue was likely twofold: the repetitive nature of social media as well as some potential underlying bias in the included parsing rules. This meant that outperforming a simple frequency test (i.e. ZeroR) was difficult to achieve.

References

[1] “Data Growth, Business Opportunities, and the IT Imperatives.” Apr. 2014. http://www.emc.com/

[2] Doug Downey. Lecture. Northwestern University. Evanston. Mar. 29, 2016.

[3] Gale, Church, and David Yarowsky. “One Sense Per Discourse. ” http://www.aclweb.org/anthology/H92-1045

[4] David Yarowsky. “One Sense Per Collocation.” http://www.aclweb.org/anthology/H93-1052

[5] Kevin Atkinson “SCOWL (and friends),” http://wordlist.aspell.net. Apr. 14, 2016.