ENG385: Paper 2
Data Modeling Assignment

Eddie Zhou

Due: 3/21/16, 5:00pm

1 Introduction

For this assignment, we choose to use programmatic techniques to extract insights and augment our tag categorization process, rather than the rigid OpenRefine tool. We rely heavily on the work of Fellbaum (1998) in creating Wordnet, a large lexical database that contains words interlinked by conceptual-semantic and lexical relations, as well as the Natural Language Toolkit, a Python module `nltk` used for natural language processing, and outlined by Bird et al. (2009).

Roughly 1000 lines of code are available in the form of various Python scripts in a public Github repository, `eng385_paper2`. The Github repository also includes a readme that, in conjunction with the code comments, outlines more explicitly the technical implementation steps, while this paper gives a higher level explanation.

The rest of this paper is organized as follows: we outline the cleaning portion of our data pipeline in section 2. Section 3 explains how we use Wordnet to structure our investigation of the categories, which are outlined in Section 4. Within Section 4, we outline the top/subcategorization methods, the more advanced of which is our clustering heuristic. Finally, in Section 5, we discuss the resulting top-level and subcategories, difficulties in the process, as well as potential for future work and improvement.

2 Cleaning

We read in the original data source, `book-tag-all.csv` as a `pandas` DataFrame and apply standard NLP normalization. This includes any cleaning that is believed to generally preserve meaning while decreasing the size of the set of unique tags. The steps include removing leading and trailing whitespace, removing non-alphabet character, converting internal whitespace to underscores, and lowercasing. For example, the cleaned version of the raw tag " `Acciden5t prevention` " would be "`accident_prevention`". Note that this results in multi-word phrases, such as " `american_sign_language`".

As outlined by Manning et al. (2009), natural text contains words that appear in different grammatical forms (e.g. organize, organizes, and organizing), as well as derivationally-related families (e.g. democracy, democratic, and democratization). In order to again decrease the size of the unique set of tags while preserving semantic meaning, we apply lemmatization to reduce words to a common base form. There are a variety of algorithms for lemmatization, and we opt to use the WordNet Lemmatizer. A simple example of the result of this data transformation is the removing of the pluralization: "`trees`" becomes "`tree`".

3 Analysis

After normalization and lemmatization, the number of unique tags decreases from **9226** to **8455**. In the pre-labeling analysis, we largely use this set, rather than the full tagset of 54787, as the duplicated tags provide the same semantic value as de-duped versions. Before outlining our analysis rationale, however, we give a brief overview of the structure and terminology of Wordnet.

Wordnet relies largely on "synsets", which are sets of cognitive synonyms. Given raw text, Wordnet can provide a set of synsets, where each synset contains a name, a part of speech, and an ID (`fall.n.01` vs `fall.v.01`). It is these synsets that are interlinked within Wordnet by IsA relationships (hypernomy and hyponomy), which point from general synsets (`furniture`) to specific ones (`couch`). In this case, `furniture` might be a hypernym of `couch`.

Our first instinct in defining categories was to find the highest level (most upper-level hypernyms) and use the top 10 highest synsets to structure our categories. After investigating the distribution of depths, we realiezd that different parts of speech have different organization. While nouns and verbs have real IsA and KindOf relations, adjectives and adverbs have "similar to" relations and not hyper/hyponyms. Moreover, after collecting the set of all root hypernyms for our verb tags, we found 430 unique verb synsets, which is difficult to reduce to a smaller number of categories. Fortunately, the noun hierarchy is more centralized, with a singular root `entity.n.01`, which has three hyponyms: `abstraction.n.06`, `physical_entity.n.01`, and `thing.n.08`. Examining our specific tag data, all tags that have noun synsets point to one of the first two root hyponyms (there are no `thing.n.08`'s).

With this information, we note that at least four of our top-level categories should be noun, verb, adjective, and adverbs. We also, however, note the presence of tags that do not fit neatly into these categories – namely, phrases and single letters. Moreover, given that the semantic meaning of descriptive adjectives can be heavily influenced by sentiment, we felt that positive and negative sentiments for adjectives is a categorization worth exploring. The final categories and their subcategories are detailed in the next section (ordered generally based on difficulty of technical implementation).

4 Categorization

1. Phrase (2-word / 3-word / 4+-word)
2. Letter (a-i / j-r / s-z)
3. Other (real word, no synset / slang / typo)
4. Positive Adjective (cluster 1 / cluster 2 / cluster 3)
5. Negative Adjective (cluster 1 / cluster 2 / cluster 3)
6. Neutral Adjective (cluster 1 / cluster 2 / cluster 3)
7. Noun (Abstraction) (cluster 1 / cluster 2 / cluster 3)
8. Noun (Physical Entity) (cluster 1 / cluster 2 / cluster 3)
9. Adverb (Time + Space + Frequency / Manner / Degree)
10. Verb (cluster 1 / cluster 2 / cluster 3)

Categories 1 and 2 and their subcategories are self-explanatory. Category 3 represents tags that have no Wordnet synsets, but are phrases or single letters. A Category-3 tag is then looked up in an English dictionary – using `pyenchant` by Kelly (1990-2015) – and labeled as subcategory 1 if it is a valid English word. We then check it against a list of slang which is programmatically web-scraped from a site maintained by Rader (1996-2016) and give it subcategory 2 if it is contained in this list. If it fits in none of these subcategories, we categorize it as a typo, subcategory 3.

Categorization into 4-10 require determining a singular synset for a given tag. Because a synset contains both a part of speech and an identifier, we must determine how to select both. The Wordnet manual tells us that the identifiers are ordered generally by usage (`chair.n.01` is more common than `chair.n.05`), so we can deterministically select the lowest identifier each time. Part-of-speech selection, however, requires a more complex process.

To pick a part of speech for a given tag's set of synsets with theoretical rigor, we make use of empirical frequency distributions. For each of the text transcriptions, we apply the contextual POS tagger available in `nltk`. The tagger examines a list of tokens (sentences with stop words such as "the" and "a" removed), and given contextual features and previous training on a corpus, tags each token as a POS (such as NN – common noun, NNP – proper noun, VB – base verb, VBG – present participle / gerund verb, etc.). Then, we construct both a conditional frequency distribution and a frequency distribution. The CFD tells us that given a word, it has appeared as "NN" x times, "VB" y times, and so on. The FD tells us that there are w total tokens tagged as "NN", z total tokens tagged as "VB", and so on. We form these distributions for

each text transcription, and then transform them into probability distributions. A simple example: if our frequency distribution is "NN": 12, "VB": 8, then our probability distribution will be "NN": 0.6, "VB" 0.4. This simply means that basd on the text, for a random word, there is a 60% chance it will be "NN", and a 40% chance it will be "VB". Note that for our purposes, we combine all broader POS forms into a single tag for both nouns and verbs ('NN' and 'NNP' both become "n") – this allows us to go between the synset single-word tag ("n" from `chair.n.01`) to the frequency distributions.

We can then go through each tag and look up the text it came from. If the tag actually appeared in the text, we can sample from its conditional probability distribution to select its synset. If it does not, we use the overall probability distribution of POS in that text. We must also renormalize the distributions based on the set of synset single-word tags – for example, if "climb" gives synsets `climb.v.01`, `climb.n.01`, and the probability distribution we use has "n": 0.5, "v": 0.4, "a": 0.1, we will use the renormalized distribution "n": 0.56, "v": 0.44, because there is no synset of climb that is an adjective synset. Note that this results in a repeated tag for the same book having potentially different categorizations, which is acceptable from a statistical sampling standpoint.

Now that we have selected a POS and a single synset for each tag, it is easy to check for an adjective tag's membership in positive and negative sentiment lists – we use one compiled by Breen (2011). Adjective tags are labeled as Category 6 if they are in neither of those lists. For the noun tags, we can easily determine whether its non-entity root is `abstraction.n.06` or `physical_entity.n.01`, and the verbs and adverbs we only give one top-level category each. The subcategories for categories 4-8 and 10 are done using our clustering heuristic, which is outlined in the subsection below. The adverb category (Category 9) is manually subcategorized (as there are only 42 unique adverbs) according to a combination of types of English adverbs as described by MyEnglishGrammar.com (2016): time/space/frequency, manner, and degree.

4.1 Clustering

We initially looked to apply a version of the classical K-Means clustering algorithm, using a negative loss function applicable to Wordnet synsets, such as the Wu-Palmer similarity score proposed by Wu and Palmer (1994), which provides a Wordnet-based scoring of similarity between synsets (1 being the same synset, 0 being no similarity). The recentering of each cluster, however, is not computationally feasible. Therefore, we use a simple heuristic that takes advantage of random initialization.

For some large number of iterations N, randomly pick $k = 3$ synsets in our subset as centers, and assign all other synsets to the center with the largest Wu-Palmer similarity score. Each initialization, record the proportional split between the cluster assignments, as well as the aggregate Wu-Palmer similarity score. Then, choose the initialization that minimizes the mean absolute error between the actual cluster sizes and a theoretical even split. For example, if one initialization gave us clusters of size 400, 390, and 410, this would be preferable to an initialization that gave cluster sizes of 1000, 150, 50. We also verify that the initialization maximizing the evenness of the split achieves a reasonable aggregate Wu-Palmer similarity score. Picking this as a criteria points to a shift in the goal towards focusing on creating subcategories that are meaningful due to their even sizes, while still maintaining adherence to the Wu-Palmer metric.

The above method is implemented for each of the two noun categories and the verb category. For the adjective synsets, due to a lack of hypernomy structure, Wu-Palmer similarity scores is not defined. Therefore, we take the definitions from adjective synsets, extract noun tokens, and compute a normalized Wu-Palmer using these noun sets. For example, to compute our definitional Wu-Palmer score for "happy" and "sad", we would first extract the definitions "enjoying or showing or marked by joy or pleasure" and "experiencing or showing sorrow or unhappiness". Then, applying similar cleaning methods to those mentioned in Section 2, we compute Wu-Palmer scores for (joy, sorrow), (joy, unhappiness), (pleasure, sorrow), and (pleasure, unhappiness), summing the scores and dividing by 4 to obtain the normalized score. Note that this relies on the reasonable assumption that the definition of an adjective contains nouns that are semantically related.

5 Discussion

We also note that for some of the subcategories chosen reflect the rigidity of the 10 top-level and 3 subcategories criterion, and have much less meaning (semantic or otherwise) than the rigorously achieved subcategories (clustering heuristic, adverb types, etc.). First and foremost, we believe that relaxing this 10/3 criterion would enable a more sensible hierarchy.

As can be seen by counting the Unique Category column of the results, which assigns a unique code (e.g. 9-1) to a tag, there is a steady decrease from the most frequent category (8-2, or Physical Entity Nouns Cluster 2) to the least frequent (9-3, or Adverbs Cluster 3). We note that lemmatized adverbs are also relatively infrequent given that they are largely derivational forms of adjectives. Also, removal of pluralization throws away a valuable indicator for noun / verb identification ("falls" may be more frequently used as a verb than a plural noun). These point to one area for future work – our cleaning process could be refined to leave more semantic meaning at the pre-categorization level.

The phrase category is another place where data refinement improvements could provide more insight. A phrase such as "American Sign Language" is clearly a proper noun, but our cleaning and categorization process gives it a "phrase" label, which is much less meaningful. Aggregating methods could be useful even if the phrases are split into words and scored individually. We found these phrases, as well as tags with no synsets, the most difficult to analyze. If an updated vocabulary was pushed to Wordnet, it's possible that the number of Category 3 tags would decrease.

Another extension that would enable better understanding of the subcategories is through some hypernym search. For example, with our current results, we know that there are 3 subcategories in the Noun Abstraction top-level category, and each of the subcategories represents a cluster achieved through our heuristic. We do have the 3 synsets used for centers in that initialization (viewable by loading any of the clustering pickle files), but these are not necessarily representative of each cluster. An interesting extension would be to look for the highest hypernym within each cluster, which would effectively give each cluster a "name" of sorts. We conclude by nothing that there is an abundance of potential extensions of this work; we merely lay the groundwork for providing a rigorous statistical and NLP-based approach to tag categorization.

6 Honor Code

I pledge my honour that this paper represents my own work in accordance with University regulations.

References

Bird, S., Loper, E. and Klein, E. (2009). *Natural Language Processing with Python*. OReilly Media Inc.

Breen, J. (2011). Twitter sentiment analysis tutorial. https://github.com/jeffreybreen/twitter-sentiment-analysis-tutorial-201107/tree/master/data/opinion-lexicon-English.

Fellbaum, C. (1998). *WordNet: An Electronic Lexical Database*. MIT Press.

Kelly, R. (1990-2015). pyenchant. https://pypi.python.org/pypi/pyenchant/.

Manning, C. D., Raghavan, P. and Schtze, H. (2009). *An Introduction to Information Retrieval*. Cambridge University Press.

MyEnglishGrammar.com (2016). Types of adverbs. http://www.myenglishgrammar.com/lesson-4-adverbs/1-types-of-adverbs.html.

Rader, W. (1996-2016). The online slang dictionary. http://onlineslangdictionary.com/word-list/.

Wu, Z. and Palmer, M. (1994). Verb semantics and lexical selection. Association for Computational Linguistics.