Using Paraphrases to Cluster and Order Adjectives by Intensity: Phenomenal == absolutely amazing, so phenomenal > amazing

Anonymous NAACL submission

Abstract

Adjectives like *fine*, *good*, *great*, and *phenomenal* all describe quality, but differ in intensity (i.e., *fine* < *good* < *great* < *phenomenal*). Understanding these intensity differences is a necessary part of reasoning about natural language. We present a method to automatically learn the relative relationship of scalar adjectives. Our approach is based on pairwise adjective intensity relationships that are inferred by analyzing pairs of adjectival paraphrases from the Paraphrase Database (http://paraphrase.org). We propose a method to construct clusters of adjectives pertaining to a single attribute (e.g. food quality), and to rank adjectives within clusters according to their relative intensities. We apply the method to predicting the sentiment polarity of Twitter posts.

1 Introduction

Semantically similar adjectives are rarely fully interchangeable in context. For example, although *good* and *great* are synonyms, *the cookies were good* does not imply that *the cookies were great*. In fact, in the case of *fine* and *outstanding*, *the cookies were outstanding* implies quite a different sentiment than *the cookies were fine*. *Fine*, *good*, *great*, and *outstanding* are not interchangeable because they differ in intensity; a native English speaker understands that *fine* < *good* < *great* < *outstanding*.

Knowing the scalar ordering of adjectives has several applications. For example, adult language learners may struggle to learn subtle semantic differences between similar adjectives, and software for language learning could include graphical representations of scalar adjective scales to facilitate lexical acquisition (**?**). Scalar adjective intensity rankings are also potentially useful for several NLP tasks, including question answering (**?**), and textual inference, including recognizing textual entailment (**?**). In this work, we demonstrate their potential to improve scores on the task of sentiment analysis (**?**).

The relative intensities of adjectives are not included in current lexical resources. Both thesauri and WordNet (**?**) include specific types of semantic relations, including synonymy, antynomy, and hyponymy. Vector-based representations of word semantics (e.g., `word2vec` (**?**)) capture a broader notion of semantic similarity between adjectives. None of these resources, however, provide the relative intensity of pairs or sets of adjectives.

Our work is divided into two tasks: (1) Retrieve clusters of adjectives that belong on a single intensity scale, given a seed (*adjective,noun*) pair to guide the search, and (2) order adjectives within each cluster by relative intensity. We approach both tasks by inferring relationships between adjectives by analyzing paraphrases from the Paraphrase Database (PPDB) (**?**). For example, if *very good* is a paraphrase for *great* in PPDB, this is evidence that *good* and *great* modify the same attribute and therefore belong on the same intensity scale. Further, since *very* intensifies adjectives that it modifies, we infer that *good* is less intense than *great*. We construct a graph that encodes these pairwise relationships, and use it as the basis for clustering adjectives by attribute and ordering them by intensity.

Previous work has focused on clustering semantically similar adjectives (**??**). Our approach is unique in that it uses paraphrases to infer pairwise intensity relationships between adjectives and in that it uses an explicit graph structure to represent all known relationships. Previous work (**???**) has also focused on ranking sets of semantically similar adjectives by intensity.

We evaluate our adjective clusters and rankings against ground-truth clusters and rankings.

We construct the ground-truth clusters using human judgements gathered from Amazon Mechanical Turk (MTurk) HITs, and borrow the ground-truth rankings from a previously-published dataset (**?**). Then, we evaluate our system extrinsically by showing how it can be useful in the task of aspect-based sentiment analysis (**??**), where the goal is to identify the sentiment expressed for specific aspects of items mentioned in a product review.

2 Related work

2.1 Clustering semantically similar adjectives

? describe an algorithm for clustering adjectives that extracts syntactically related words from a large, parsed corpus and quantifies the similarity and dissimilarity of each pair of evaluated adjectives. Clusters are constructed by initially selecting a random partition and improving the partition iteratively using an objective function that considers the similarity and dissimilarity of clusters' members.

? use the *k*-means clustering method to generate adjective clusters. Their distance metric is cosine similarity between `word2vec` vectors.

Both **?** and **?** use "hard" clustering algorithms – algorithms that produce disjoint clusters, thus preventing words with multiple meanings from appearing in multiple clusters. By contrast, "soft" clustering algorithms accommodate polysemous words. **?** explicitly leave deriving a soft clustering approach as a separate research problem.

2.2 Ranking adjectives by intensity

? use scalar relationships between adjectives to infer the answers to polar questions that do not explicitly contain a "yes" or "no" word (e.g., *Do you think it's a good idea? I think it's an excellent idea.*). Thus, their interest is primarily in scalar relationships between pairs of adjectives. The relative intensity of adjectives is inferred by finding the adjectives in a corpora of online reviews and comparing the reviews' associated numerical ratings.

? infer pairwise intensity relationships using linguistic patterns found in a large corpus and then determine a global ordering of adjectives using a Mixed Integer Linear Program (MILP). Frequency information for the MILP is gathered from the Google N-Grams dataset. They use adjective clusters based on WordNet *dumbbells*, which each consist of two antonyms (the poles) and several other adjectives that are semantically similar to one of the poles. **?** also apply **?**'s MILP implementation, gathering frequency data from the PubMed corpus to rank the adjectives in their clusters.

particularly pleased	$\leftrightarrow$	*ecstatic*
quite limited	$\leftrightarrow$	*restricted*
rather odd	$\leftrightarrow$	*crazy*
so silly	$\leftrightarrow$	*dumb*
completely mad	$\leftrightarrow$	*crazy*

Figure 1: Sample paraphrases in PPDB

3 Constructing an adjective graph

Our basis for extracting same-aspect adjectives and ordering them by intensity is a graph of adjectives and their relationships. In the graph, which we call *JJGraph*, nodes are adjectives, and edges connect adjectives that become paraphrases when an adverb is prepended to one adjective in the pair. Here we discuss *JJGraph*'s construction and attributes.

3.1 Data: Paraphrase Database

JJGraph is built from paraphrases in the Paraphrase Database (PPDB) (**?**). Each pair of connected nodes corresponds to an adjectival paraphrase (e.g., *very tall* $\leftrightarrow$ *large*), where one 'phrase' is a single-word adjective and the other is an multi-word adjectival phrase containing an adverb and adjective. Figure **??** gives examples of paraphrase pairs encoded in the graph. In total, we use 36,756 adjective paraphrases that contain 21,701 unique adjectives and 610 unique adverbs.

3.2 Adverbs

Our method uses adjective paraphrases from PPDB that contain intensifying adverbs. An **intensifying adverb** semantically strengthens the adjective that it modifies. For example, *very* and *so* are intensifying adverbs. In contrast, a **de-intensifying adverb** semantically weakens the adjective that it modifies. For example, *slightly* and *somewhat* are de-intensifying adverbs.

We use paraphrases to make pairwise intensity relations. For example, since *so wonderful* is a paraphrase for *brilliant*, we infer that *wonderful* $<$ *brilliant*.

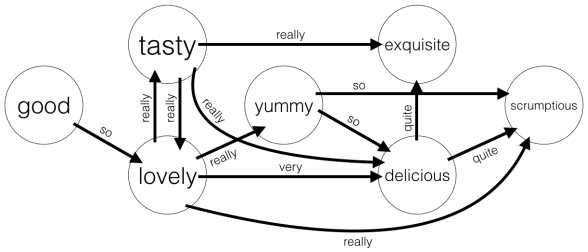

Figure 2: Directed graph structure

Round 1	**very**	hard	⇔	harder
	kinda	hard	⇔	harder
	so	hard	⇔	harder
	pretty	hard	⇔	harder
	⇓			
Round 2	very	*pleasant*	⇔	*delightful*
	kinda	*hard*	⇔	*tricky*
	so	*wonderful*	⇔	*brilliant*
	pretty	*simple*	⇔	*plain*
	⇓		⇓	
Round 3	**more**	pleasant	⇔	delightful
	really	hard	⇔	tricky
	truly	wonderful	⇔	brilliant
	fairly	simple	⇔	plain

Figure 3: Iterative process for identifying adverbs

3.3 Identifying intensifying adverbs

We used an iterative process to identify intensifying adverbs (Figure 3), and then used paraphrases containing the identified intensifying adverbs to construct *JJGraph*.

First, we identified pairs of phrases in which one phrase contained the base form of an adjective (e.g., *hard*) and the other phrase contained either the comparative or superlative form of the same adjective (e.g., *harder* or *hardest*). Such pairs were identified by lemmatizing the longer word in the pair (*harder*), and comparing the lemma (*hard*) to the shorter word in the pair (*hard*) for equality. NLTK's `WordNetLemmatizer` was used for lemmatizing (**?**).

By definition, the base form of an adjective is less semantically intense than both its comparative and superlative forms (e.g., *hard* < *harder* < *hardest*). Thus, the adverb that precedes the base form of the adjective is presumed to be an intensifying adverb.

Next, we found pairs of phrases that included the adverbs found in Round 1. For example, whereas in Round 1 we identified *very* and *pretty* as intensifying adverbs, we found in Round 2 that *pleasant* and *delightful* and that *simple* and *plain* were also related by *very* and *pretty*, respectively. That is, *pleasant* < *delightful* and *simple* < *plain*.

Finally, in Round 3, we identified additional intensifying adverbs by finding pairs of phrases with the words identified in Round 2. For example, in Round 2 we found the patterns *[adverb] pleasant* ↔ *delightful* and *[adverb] simple* ↔ *plain*, so in Round 3 we found all other pairs of phrases that fit those patterns (e.g., *more pleasant* ↔ *delightful*, *fairly simple* ↔ *plain*). The adverbs in these phrases (e.g., *more*, *fairly*) are intensifying.

In total, we identified 677 intensifying adverbs using this process. We then used the set of intensifying adverbs to search for suitable paraphrases to include in *JJGraph*.

3.4 Choosing a graph representation

The transitivity of scalar relationships (e.g., if $A < B$ and $B < C$, then $A < C$) made a graph structure a natural representation of our inferred pairwise intensity relationships. As scalar relationships inherently have a direction, the graph is directed. Additionally, the graph is a multigraph because there are frequently multiple intensifying relationships between pairs of adjectives. For example, the paraphrases *pretty hard* ↔ *tricky* and *really hard* ↔ *tricky* are both present in PPDB.

We construct a directed graph from all pairwise intensity relations. Each vertex in the graph is an adjective, and each directed edge is an intensifying relationship between two adjectives. For example, if the directed edge *(hard, tricky)* is labeled with *pretty*, then one of the original paraphrase pairs in PPDB was *pretty hard* ↔ *tricky* (Figure 2). The edges also point in the direction of increasing intensity. That is, an edge going from *hard* to *tricky* implies that *hard* < *tricky*. We refer to this graph as *JJGraph*.

3.5 Removing noise from graph representation

While experimenting with graph-based clustering algorithms, we observed that poor paraphrase edges in the graph were creating "shortcuts" in the graph (Figure 4). For example, edges connecting *best* to *largest* and to *more* resulted in *best* being poorly clustered with *largest*, *more*, *increasing*, and *major*. We decided to prune JJGraph's poorest edges.

4 Clustering adjectives in the graph

Our first goal is to retrieve clusters of adjectives from *JJGraph* that describe the same attribute as an (*adjective, noun*) pair given as input. For example, given the pair (*hot, coffee*), correct output would be a set of adjectives describing liquid temperature, such as *warm*, *lukewarm*, and *scalding*.

The reason that we require an (*adjective*, *noun*) pair as input, rather than simply an *adjective*, is that many adjectives are ambiguous. *Hot*, for example, can describe temperature, but can also be used to describe appearance or popularity. Specifying a noun as part of the input serves to disambiguate the adjective.

Our adjective extraction process consists of three steps: first, filtering adjectives connected to the input adjective in *JJGraph*; second, clustering the connected adjectives to disambiguate sense; and third, selecting the "best-fit" cluster given the input noun. Here we describe each step in more detail.

4.0.1 Filtering connected adjectives

Given an input (*adjective*, *noun*) pair, call it (j, n), our first step extracts adjectives from *JJGraph* that (a) are directly connected to the adjective j by a single (undirected) edge in *JJGraph*, and (b) have been used to modify the noun n in a corpus. The first criterion assumes that directly-connected adjectives can be used to describe the same attribute. The second criterion serves to filter out adjectives that do not describe the same attribute.

The corpus we use for this step and the next is a set of business reviews from the Yelp Dataset Challenge (**?**). This set of 2.6M reviews covering 86K businesses is useful because it contains many instances of adjectives of varying intensity that modify a single attribute (such as food taste, or service speed). When filtering adjectives connected to j in *JJGraph*, we retain all directly-connected j' that have modified n at least ten times in the Yelp corpus, and are also highly associated with n. We quantify the level of association between an adjective-noun pair (j, n) using (normalized) pointwise mutual information $nmi_{j,n}$, with a discounting factor $dsc_{j,n}$ to down-weight highly infrequent pairs (**?**):

$$nmi_{j,n} = -\log_2(mi_{j,n} \times dsc_{j,n})/nrm_{j,n}$$

$$mi_{j,n} = \frac{\frac{f(j,n)}{N}}{\frac{\sum_u f(u,n)}{N} \times \frac{\sum_v f(j,v)}{N}}$$

$$dsc_{j,n} = \frac{f(j,n)}{f(j,n)+1} \times \frac{min(\sum_u f(u,n), \sum_v f(j,v))}{min(\sum_u f(u,n), \sum_v f(j,v)) + 1}$$

$$nrm_{j,n} = -\log_2 \frac{f(j,n)}{N}$$

where $N = \sum_u \sum_v f(u,v)$ is the total frequency count of all adjective-noun pairs in the Yelp corpus.

When filtering adjectives, we require each j' to have $PMI(', n) > t$, where t is a threshold parameter. In our experiments we vary $t \in \{0.0, 0.01, 0.1\}$.

4.0.2 Clustering filtered adjectives

Even after filtering adjectives to include only those known to modify the input noun, it is still possible to retrieve a set of adjectives describing multiple attributes. This is particularly true when the input adjective is polysemous. For example, the set of filtered adjectives returned for the query (*hot,coffee*) includes *fantastic, fabulous, spectacular*, and *cold, lukewarm*, although only the last two are applicable in most contexts.

To counteract this result, we cluster the set of filtered adjectives by the sense of the target they convey. Using *JJGraph* this is a simple process; we simply take the subgraph formed by removing all nodes from *JJGraph* except the set of filtered adjectives, and partition the resulting subgraph into connected components. This serves to divide the filtered adjectives into coarse sense clusters.

4.0.3 Selecting the best cluster

Given a set of one or more clusters of adjectives, the last step is to choose which cluster is most applicable to the original adjective-noun input pair. The most applicable cluster should contain adjectives j' that are (a) highly similar to the input adjective, j, and (b) highly associated with the input noun, n. Therefore we assign a score s_c to each cluster, where

$$s_C = avg_{j' \in C} nmi_{j',n} \times avg_{j' \in C} PPDBScore_{j,j'}$$

The mutual information, $nmi_{j',n}$ is calculated as before. The component $PPDBScore_{j,j'}$ gives the PPDB 2.0 Score between adjectives.[1]

[1] PPDB2.0 Score is a supervised metric designed to correlated with human judgements of paraphrase quality (**?**)

Method	Query	Result

Table 1: Clusters retrieved for various queries by our method.

The final cluster $\hat{C}$ returned by our algorithm is that which produces the maximum s_C.

4.1 Qualitative Evaluation

We qualitatively examine the adjective clusters retrieved by our methods.

5 Ordering adjectives by intensity

6 Application: Predicting aspect-based sentiment polarity

7 Conclusion

In this work, we have presented our progress towards developing a pipeline to cluster semantically similar adjectives and to rank adjectives within a cluster according to their relative intensity. Future work will include further experimentation with clustering algorithms and developing algorithms to rank related adjectives within a cluster on a scale