

Aspect Term Extraction for Sentiment Analysis: New Datasets, New Evaluation Measures and an Improved Unsupervised Method

John Pavlopoulos and Ion Androutsopoulos

Dept. of Informatics, Athens University of Economics and Business, Greece

<http://nlp.cs.aueb.gr/>

Abstract

Given a set of texts discussing a particular entity (e.g., customer reviews of a smart-phone), aspect based sentiment analysis (ABSA) identifies prominent aspects of the entity (e.g., battery, screen) and an average sentiment score per aspect. We focus on aspect term extraction (ATE), one of the core processing stages of ABSA that extracts terms naming aspects. We make publicly available three new ATE datasets, arguing that they are better than previously available ones. We also introduce new evaluation measures for ATE, again arguing that they are better than previously used ones. Finally, we show how a popular unsupervised ATE method can be improved by using continuous space vector representations of words and phrases.

1 Introduction

Before buying a product or service, consumers often search the Web for expert reviews, but increasingly also for opinions of other consumers, expressed in blogs, social networks etc. Many useful opinions are expressed in text-only form (e.g., in tweets). It is then desirable to *extract aspects* (e.g., screen, battery) from the texts that discuss a particular entity (e.g., a smartphone), i.e., figure out what is being discussed, and also *estimate aspect sentiment scores*, i.e., how positive or negative the (usually average) sentiment for each aspect is. These two goals are jointly known as *Aspect Based Sentiment Analysis* (ABSA) (Liu, 2012).

In this paper, we consider free text customer reviews of products and services; ABSA, however, is also applicable to texts about other kinds of entities (e.g., politicians, organizations). We assume that a search engine retrieves customer reviews about a particular *target entity* (product or

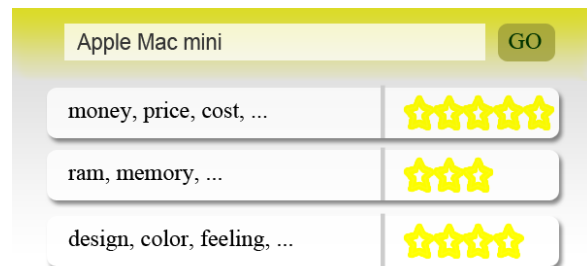


Figure 1: Automatically extracted prominent aspects (shown as clusters of aspect terms) and average aspect sentiment scores of a target entity.

service), that multiple reviews written by different customers are retrieved for each target entity, and that the ultimate goal is to produce a table like the one of Fig. 1, which presents the most prominent aspects and average aspect sentiment scores of the target entity. Most ABSA systems in effect perform all or some of the following three subtasks:

Aspect term extraction: Starting from texts about a particular target entity or entities of the same type as the target entity (e.g., laptop reviews), this stage extracts and possibly ranks by importance *aspect terms*, i.e., terms naming aspects (e.g., ‘battery’, ‘screen’) of the target entity, including multi-word terms (e.g., ‘hard disk’) (Liu, 2012; Long et al., 2010; Snyder and Barzilay, 2007; Yu et al., 2011). At the end of this stage, each aspect term is taken to be the name of a different aspect, but aspect terms may subsequently be clustered during aspect aggregation; see below.

Aspect term sentiment estimation: This stage estimates the polarity and possibly also the intensity (e.g., strongly negative, mildly positive) of the opinions for each aspect term of the target entity, usually averaged over several texts. Classifying texts by sentiment polarity is a popular research topic (Liu, 2012; Pang and Lee, 2005; Tsytsarau and Palpanas, 2012). The goal, however, in this

ABSA subtask is to estimate the (usually average) polarity and intensity of the opinions about particular aspect terms of the target entity.

Aspect aggregation: Some systems group aspect terms that are synonyms or near-synonyms (e.g., ‘price’, ‘cost’) or, more generally, cluster aspect terms to obtain aspects of a coarser granularity (e.g., ‘chicken’, ‘steak’, and ‘fish’ may all be replaced by ‘food’) (Liu, 2012; Long et al., 2010; Zhai et al., 2010; Zhai et al., 2011). A polarity (and intensity) score can then be computed for each coarser aspect (e.g., ‘food’) by combining (e.g., averaging) the polarity scores of the aspect terms that belong in the coarser aspect.

In this paper, we focus on aspect term extraction (ATE). Our contribution is threefold. Firstly, we argue (Section 2) that previous ATE datasets are not entirely satisfactory, mostly because they contain reviews from a particular domain only (e.g., consumer electronics), or they contain reviews for very few target entities, or they do not contain annotations for aspect terms. We constructed and make publicly available three new ATE datasets with customer reviews for a much larger number of target entities from three domains (restaurants, laptops, hotels), with gold annotations of all the aspect term occurrences; we also measured inter-annotator agreement, unlike previous datasets.

Secondly, we argue (Section 3) that commonly used evaluation measures are also not entirely satisfactory. For example, when precision, recall, and F -measure are computed over *distinct* aspect terms (types), equal weight is assigned to more and less frequent aspect terms, whereas frequently discussed aspect terms are more important; and when precision, recall, and F -measure are computed over aspect term *occurrences* (tokens), methods that identify very few, but very frequent aspect terms may appear to perform much better than they actually do. We propose weighted variants of precision and recall, which take into account the *rankings* of the distinct aspect terms that are obtained when the distinct aspect terms are ordered by their true and predicted frequencies. We also compute the average weighted precision over several weighted recall levels.

Thirdly, we show (Section 4) how the popular *unsupervised* ATE method of Hu and Liu (2004), can be extended with continuous space word vectors (Mikolov et al., 2013a; Mikolov et al., 2013b; Mikolov et al., 2013c). Using our datasets and

evaluation measures, we demonstrate (Section 5) that the extended method performs better.

2 Datasets

We first discuss previous datasets that have been used for ATE, and we then introduce our own.

2.1 Previous datasets

So far, ATE methods have been evaluated mainly on customer reviews, often from the consumer electronics domain (Hu and Liu, 2004; Popescu and Etzioni, 2005; Ding et al., 2008).

The most commonly used dataset is that of Hu and Liu (2004), which contains reviews of only five particular electronic products (e.g., Nikon Coolpix 4300). Each sentence is annotated with aspect terms, but inter-annotator agreement has not been reported.¹ All the sentences appear to have been selected to express clear positive or negative opinions. There are no sentences expressing conflicting opinions about aspect terms (e.g., “The screen is clear but small”), nor are there any sentences that do not express opinions about their aspect terms (e.g., “It has a 4.8-inch screen”). Hence, the dataset is not entirely representative of product reviews. By contrast, our datasets, discussed below, contain reviews from three domains, including sentences that express conflicting or no opinions about aspect terms, they concern many more target entities (not just five), and we have also measured inter-annotator agreement.

The dataset of Ganu et al. (2009), on which one of our datasets is based, is also popular. In the original dataset, each sentence is tagged with coarse aspects (‘food’, ‘service’, ‘price’, ‘ambience’, ‘anecdotes’, or ‘miscellaneous’). For example, “The restaurant was expensive, but the menu was great” would be tagged with the coarse aspects ‘price’ and ‘food’. The coarse aspects, however, are not necessarily terms occurring in the sentence, and it is unclear how they were obtained. By contrast, we asked human annotators to mark the explicit aspect *terms* of each sentence, leaving the task of clustering the terms to produce coarser aspects for an aspect aggregation stage.

The ‘Concept-Level Sentiment Analysis Challenge’ of ESWC 2014 uses the dataset of Blitzer et al. (2007), which contains customer reviews of

¹Each aspect term occurrence is also annotated with a sentiment score. We do not discuss these scores here, since we focus on ATE. The same comment applies to the dataset of Ganu et al. (2009) and our datasets.

DVDs, books, kitchen appliances, and electronic products, with an overall sentiment score for each review. One of the challenge’s tasks requires systems to extract the aspects of each sentence and a sentiment score (positive or negative) per aspect.² The aspects are intended to be concepts from ontologies, not simply aspect terms. The ontologies to be used, however, are not fully specified and no training dataset with sentences and gold aspects is currently available.

Overall, the previous datasets are not entirely satisfactory, because they contain reviews from a particular domain only, or reviews for very few target entities, or their sentences are not entirely representative of customer reviews, or they do not contain annotations for aspect terms, or no inter-annotator agreement has been reported. To address these issues, we provide three new ATE datasets, which contain customer reviews of restaurants, hotels, and laptops, respectively.³

2.2 Our datasets

The restaurants dataset contains 3,710 English sentences from the reviews of Ganu et al. (2009).⁴ We asked human annotators to tag the aspect terms of each sentence. In “The *dessert* was divine”, for example, the annotators would tag the aspect term ‘dessert’. In a sentence like “The restaurant was expensive, but the *menu* was great”, the annotators were instructed to tag only the explicitly mentioned aspect term ‘menu’. The sentence also refers to the prices, and a possibility would be to add ‘price’ as an *implicit aspect term*, but we do not consider implicit aspect terms in this paper.

We used nine annotators for the restaurant reviews. Each sentence was processed by a single annotator, and each annotator processed approximately the same number of sentences. Among the 3,710 restaurant sentences, 1,248 contain exactly one aspect term, 872 more than one, and 1,590 no aspect terms. There are 593 distinct multi-word aspect terms and 452 distinct single-word aspect terms. Removing aspect terms that occur only once leaves 67 distinct multi-word and 195 distinct single-word aspect terms.

The hotels dataset contains 3,600 English sen-

tences from online customer reviews of 30 hotels. We used three annotators. Among the 3,600 hotel sentences, 1,326 contain exactly one aspect term, 652 more than one, and 1,622 none. There are 199 distinct multi-word aspect terms and 262 distinct single-word aspect terms, of which 24 and 120, respectively, were tagged more than once.

The laptops dataset contains 3,085 English sentences of 394 online customer reviews. A single annotator (one of the authors) was used. Among the 3,085 laptop sentences, 909 contain exactly one aspect term, 416 more than one, and 1,760 none. There are 350 distinct multi-word and 289 distinct single-word aspect terms, of which 67 and 137, respectively, were tagged more than once.

To measure inter-annotator agreement, we used a sample of 75 restaurant, 75 laptop, and 100 hotel sentences. Each sentence was processed by two (for restaurants and laptops) or three (for hotels) annotators, other than the annotators used previously. For each sentence s_i , the inter-annotator agreement was measured as the Dice coefficient $D_i = 2 \cdot \frac{|A_i \cap B_i|}{|A_i| + |B_i|}$, where A_i, B_i are the sets of aspect term occurrences tagged by the two annotators, respectively, and $|S|$ denotes the cardinality of a set S ; for hotels, we use the mean pairwise D_i of the three annotators.⁵ The overall inter-annotator agreement D was taken to be the average D_i of the sentences of each sample. We, thus, obtained $D = 0.72, 0.70, 0.69$, for restaurants, hotels, and laptops, respectively, which indicate reasonably high inter-annotator agreement.

2.3 Single and multi-word aspect terms

ABSA systems use ATE methods ultimately to obtain the m most prominent (frequently discussed) distinct aspect terms of the target entity, for different values of m .⁶ In a system like the one of Fig. 1, for example, if we ignore aspect aggregation, each row will report the average sentiment score of a single frequent distinct aspect term, and m will be the number of rows, which may depend on the display size or user preferences.

Figure 2 shows the percentage of distinct multi-word aspect terms among the m most frequent distinct aspect terms, for different values of m , in

²See <http://2014.eswc-conferences.org/>.

³Our datasets are available upon request. The datasets of the ABSA task of SemEval 2014 (<http://alt.qcri.org/semeval2014/task4/>) are based on our datasets.

⁴The original dataset of Ganu et al. contains 3,400 sentences, but some of the sentences had not been properly split.

⁵Cohen’s Kappa cannot be used here, because the annotators may tag any word sequence of any sentence, which leads to a very large set of categories. A similar problem was reported by Kobayashi et al. (2007).

⁶A more general definition of prominence might also consider the average sentiment score of each distinct aspect term.

our three datasets and the electronics dataset of Hu and Liu (2004). There are many more single-word distinct aspect terms than multi-word distinct aspect terms, especially in the restaurant and hotel reviews. In the electronics and laptops datasets, the percentage of multi-word distinct aspect terms (e.g., ‘hard disk’) is higher, but most of the distinct aspect terms are still single-word, especially for small values of m . By contrast, many ATE methods (Hu and Liu, 2004; Popescu and Etzioni, 2005; Wei et al., 2010) devote much of their processing to identifying multi-word aspect terms.

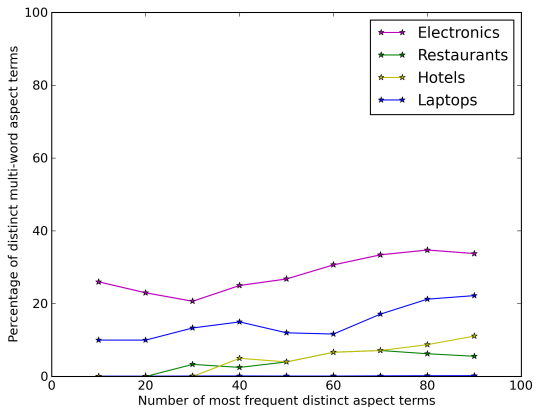


Figure 2: Percentage of (distinct) multi-word aspect terms among the most frequent aspect terms.

3 Evaluation measures

We now discuss previous ATE evaluation measures, also introducing our own.

3.1 Precision, Recall, F-measure

ATE methods are usually evaluated using precision, recall, and F -measure (Hu and Liu, 2004; Popescu and Etzioni, 2005; Kim and Hovy, 2006; Wei et al., 2010; Moghaddam and Ester, 2010; Bagheri et al., 2013), but it is often unclear if these measures are applied to *distinct* aspect terms (no duplicates) or aspect term *occurrences*.

In the former case, each method is expected to return a set A of distinct aspect terms, to be compared to the set G of distinct aspect terms the human annotators identified in the texts. TP (true positives) is $|A \cap G|$, FP (false positives) is $|A \setminus G|$, FN (false negatives) is $|G \setminus A|$, and precision (P), recall (R), $F = \frac{2 \cdot P \cdot R}{P + R}$ are defined as usually:

$$P = \frac{TP}{TP + FP}, R = \frac{TP}{TP + FN} \quad (1)$$

This way, however, precision, recall, and F -measure assign the same importance to all the distinct aspect terms, whereas missing, for example, a more frequent (more frequently discussed) distinct aspect term should probably be penalized more heavily than missing a less frequent one.

When precision, recall, and F -measure are applied to aspect term *occurrences* (Liu et al., 2005), TP is the number of aspect term occurrences tagged (each term occurrence) both by the method being evaluated and the human annotators, FP is the number of aspect term occurrences tagged by the method but not the human annotators, and FN is the number of aspect term occurrences tagged by the human annotators but not the method. The three measures are then defined as above. They now assign more importance to frequently occurring distinct aspect terms, but they can produce misleadingly high scores when only a few, but very frequent distinct aspect terms are handled correctly. Furthermore, the occurrence-based definitions do not take into account that missing several aspect term occurrences or wrongly tagging several expressions as aspect term occurrences may not actually matter, as long as the m most frequent distinct aspect terms can be correctly reported.

3.2 Weighted precision, recall, AWP

What the previous definitions of precision and recall miss is that in practice ABSA systems use ATE methods ultimately to obtain the m most frequent distinct aspect terms, for a range of m values. Let A_m and G_m be the *lists* that contain the m most frequent distinct aspect terms, ordered by their predicted and true frequencies, respectively; the predicted and true frequencies are computed by examining how frequently the ATE method or the human annotators, respectively, tagged occurrences of each distinct aspect term. Differences between the predicted and true frequencies do not matter, as long as $A_m = G_m$, for every m . Not including in A_m a term of G_m should be penalized more or less heavily, depending on whether the term’s true frequency was high or low, respectively. Furthermore, including in A_m a term not in G_m should be penalized more or less heavily, depending on whether the term was placed towards the beginning or the end of A_m , i.e., depending on the prominence that was assigned to the term.

To address the issues discussed above, we introduce weighted variants of precision and recall.

For each ATE method, we now compute a single list $A = \langle a_1, \dots, a_{|A|} \rangle$ of distinct aspect terms identified by the method, ordered by decreasing predicted frequency. For every m value (number of most frequent distinct aspect terms to show), the method is treated as having returned the sublist A_m with the first m elements of A . Similarly, we now take $G = \langle g_1, \dots, g_{|G|} \rangle$ to be the list of the distinct aspect terms that the human annotators tagged, ordered by decreasing true frequency.⁷ We define weighted precision (WP_m) and weighted recall (WR_m) as in Eq. 2–3. The notation $1\{\kappa\}$ denotes 1 if condition κ holds, and 0 otherwise. By $r(a_i)$ we denote the ranking of the returned term a_i in G , i.e., if $a_i = g_j$, then $r(a_i) = j$; if $a_i \notin G$, then $r(a_i)$ is an arbitrary positive integer.

$$WP_m = \frac{\sum_{i=1}^m \frac{1}{i} \cdot 1\{a_i \in G\}}{\sum_{i=1}^m \frac{1}{i}} \quad (2)$$

$$WR_m = \frac{\sum_{i=1}^m \frac{1}{r(a_i)} \cdot 1\{a_i \in G\}}{\sum_{j=1}^{|G|} \frac{1}{j}} \quad (3)$$

WR_m counts how many terms of G (gold distinct aspect terms) the method returned in A_m , but weighting each term by its inverse ranking $\frac{1}{r(a_i)}$, i.e., assigning more importance to terms the human annotators tagged more frequently. The denominator of Eq. 3 sums the weights of all the terms of G ; in unweighted recall applied to distinct aspect terms, where all the terms of G have the same weight, the denominator would be $|G| = TP + FN$ (Eq. 1). WP_m counts how many gold aspect terms the method returned in A_m , but weighting each returned term a_i by its inverse ranking $\frac{1}{i}$ in A_m , to reward methods that return more gold aspect terms towards the beginning of A_m . The denominator of Eq. 2 sums the weights of all the terms of A_m ; in unweighted precision applied to distinct aspect terms, the denominator would be $|A_m| = TP + FN$ (Eq. 1).

We plot weighted precision-recall curves by computing WP_m, WR_m pairs for different values of m , as in Fig. 3 below.⁸ The higher the curve of a method, the better the method. We also compute the average (interpolated) weighted precision

(AWP) of each method over 11 recall levels:

$$AWP = \frac{1}{11} \sum_{r \in \{0, 0.1, \dots, 1\}} WP_{int}(r)$$

$$WP_{int}(r) = \max_{m \in \{1, \dots, |A|\}, WR_m \geq r} WP_m$$

AWP is similar to average (interpolated) precision (AP), which is used to summarize the tradeoff between (unweighted) precision and recall.

3.3 Other related measures

Yu et al. (2011) used $nDCG@m$ (Järvelin and Kekäläinen, 2002; Sakai, 2004; Manning et al., 2008), defined below, to evaluate each list of m distinct aspect terms returned by an ATE method.

$$nDCG@m = \frac{1}{Z} \sum_{i=1}^m \frac{2^{t(i)} - 1}{\log_2(1+i)}$$

Z is a normalization factor to ensure that a perfect ranking gets $nDCG@m = 1$, and $t(i)$ is a reward function for a term placed at position i of the returned list. In the work of Yu et al., $t(i) = 1$ if the term at position i is not important (as judged by a human), $t(i) = 2$ if the term is ‘ordinary’, and $t(i) = 3$ if it is important. The logarithm is used to reduce the reward for distinct aspect terms placed at lower positions of the returned list.

The $nDCG@m$ measure is well known in ranking systems (e.g., search engines) and it is similar to our weighted precision (WP_m). The denominator of Eq. 2 corresponds to the normalization factor Z of $nDCG@m$; the $\frac{1}{i}$ factor of in the numerator of Eq. 2 corresponds to the $\frac{1}{\log_2(1+i)}$ degradation factor of $nDCG@m$; and the $1\{a_i \in G\}$ factor of Eq. 2 is a binary reward function, corresponding to the $2^{t(i)} - 1$ factor of $nDCG@m$.

The main difference from $nDCG@m$ is that WP_m uses a degradation factor $\frac{1}{i}$ that is inversely proportional to the ranking of the returned term a_i in the returned list A_m , whereas $nDCG@m$ uses a logarithmic factor $\frac{1}{\log_2(1+i)}$, which reduces less sharply the reward for distinct aspect terms returned at lower positions in A_m . We believe that the degradation factor of WP_m is more appropriate for ABSA, because most users would in practice wish to view sentiment scores for only a few (e.g., $m = 10$) frequent distinct aspect terms, whereas in search engines users are more likely to examine more of the highly-ranked returned items. It is possible, however, to use a logarithmic degradation factor in WP_m , as in $nDCG@m$.

⁷In our experiments, we exclude from G aspect terms tagged by the annotators only once.

⁸With supervised methods, we perform a 10-fold cross-validation for each m , and we macro-average WP_m, WR_m over the folds. We provide our datasets partitioned in folds.

Another difference is that we use a binary reward factor $1\{a_i \in G\}$ in WP_m , instead of the $2^{t(i)} - 1$ factor of $nDCG@m$ that has three possible values in the work of Yu et al. (2011). We use a binary reward factor, because preliminary experiments we conducted indicated that multiple relevance levels (e.g., not an aspect term, aspect term but unimportant, important aspect term) confused the annotators and led to lower inter-annotator agreement. The $nDCG@m$ measure can also be used with a binary reward factor; the possible values $t(i)$ would be 0 and 1.

With a binary reward factor, $nDCG@m$ in effect measures the ratio of correct (distinct) aspect terms to the terms returned, assigning more weight to correct aspect terms placed closer to the top of the returned list, like WP_m . The $nDCG@m$ measure, however, does not provide any indication of how many of the gold distinct aspect terms have been returned. By contrast, we also measure weighted recall (Eq. 3), which examines how many of the (distinct) gold aspect terms have been returned in A_m , also assigning more weight to the gold aspect terms the human annotators tagged more frequently. We also compute the average weighted precision (AWP), which is a combination of WP_m and WR_m , for a range of m values.

4 Aspect term extraction methods

We implemented and evaluated four ATE methods: (i) a popular baseline (dubbed **FREQ**) that returns the most frequent distinct nouns and noun phrases, (ii) the well-known method of Hu and Liu (2004), which adds to the baseline pruning mechanisms and steps that detect more aspect terms (dubbed **H&L**), (iii) an extension of the previous method (dubbed **H&L+W2V**), with an extra pruning step we devised that uses the recently popular continuous space word vectors (Mikolov et al., 2013c), and (iv) a similar extension of **FREQ** (dubbed **FREQ+W2V**). All four methods are *unsupervised*, which is particularly important for ABSA systems intended to be used across domains with minimal changes. They return directly a list A of distinct aspect terms ordered by decreasing predicted frequency, rather than tagging aspect term occurrences, which would require computing the A list from the tagged occurrences before applying our evaluation measures (Section 3.2).

4.1 The FREQ baseline

The **FREQ** baseline returns the most frequent (distinct) nouns and noun phrases of the reviews in each dataset (restaurants, hotels, laptops), ordered by decreasing sentence frequency (how many sentences contain the noun or noun phrase).⁹ This is a reasonably effective and popular baseline (Hu and Liu, 2004; Wei et al., 2010; Liu, 2012).

4.2 The H&L method

The method of Hu and Liu (2004), dubbed **H&L**, first extracts all the distinct nouns and noun phrases from the reviews of each dataset (lines 3–6 of Algorithm 1) and considers them candidate distinct aspect terms.¹⁰ It then forms longer candidate distinct aspect terms by concatenating pairs and triples of candidate aspect terms occurring in the same sentence, in the order they appear in the sentence (lines 7–11). For example, if ‘battery life’ and ‘screen’ occur in the same sentence (in this order), then ‘battery life screen’ will also become a candidate distinct aspect term.

The resulting candidate distinct aspect terms are ordered by decreasing *p-support* (lines 12–15). The *p-support* of a candidate distinct aspect term t is the number of sentences that contain t , excluding sentences that contain another candidate distinct aspect term t' that subsumes t . For example, if both ‘battery life’ and ‘battery’ are candidate distinct aspect terms, a sentence like “The battery life was good” is counted in the *p-support* of ‘battery life’, but not in the *p-support* of ‘battery’.

The method then tries to correct itself by pruning wrong candidate distinct aspect terms and detecting additional candidates. Firstly, it discards multi-word distinct aspect terms that appear in ‘non-compact’ form in more than one sentences (lines 16–23). A multi-word term t appears in non-compact form in a sentence if there are more than three other words (not words of t) between any two of the words of t in the sentence. For example, the candidate distinct aspect term ‘battery life screen’ appears in non-compact form in “battery life is way better than screen”. Secondly, if the *p-support* of a candidate distinct aspect term t is smaller than 3 and t is subsumed by another can-

⁹We use the default POS tagger of NLTK, and the chunker of NLTK trained on the Treebank corpus; see <http://nltk.org/>. We convert all words to lower-case.

¹⁰Some details of the work of Hu and Liu (2004) were not entirely clear to us. The discussion here and our implementation reflect our understanding.

didate distinct aspect term t' , then t is discarded (lines 21–23).

Subsequently, a set of ‘opinion adjectives’ is formed; for each sentence and each candidate distinct aspect term t that occurs in the sentence, the closest to t adjective of the sentence (if there is one) is added to the set of opinion adjectives (lines 25–27). The sentences are then re-scanned; if a sentence does not contain any candidate aspect term, but contains an opinion adjective, then the nearest noun to the opinion adjective is added to the candidate distinct aspect terms (lines 28–31). The remaining candidate distinct aspect terms are returned, ordered by decreasing p -support.

Algorithm 1 The method of Hu and Liu

Require: sentences: a list of sentences
1: terms = new Set(String)
2: psupport = new Map(String, int)
3: **for** s in sentences **do**
4: nouns = POSTagger(s).getNouns()
5: nps = Chunker(s).getNPChunks()
6: terms.add(nouns \cup nps)
7: **for** s in sentences **do**
8: **for** t1, t2 in terms s.t. t1, t2 in s \wedge
 s.index(t1) < s.index(t2) **do**
9: terms.add(t1 + " " + t2)
10: **for** t1, t2, t3 in s.t. t1, t2, t3 in s \wedge
 s.index(t1) < s.index(t2) < s.index(t3) **do**
11: terms.add(t1 + " " + t2 + " " + t3)
12: **for** s in sentences **do**
13: **for** t: t in terms \wedge t in s **do**
14: **if** $\neg \exists t'$: t' in terms \wedge t' in s \wedge t in t' **then**
15: psupport[term] += 1
16: nonCompact = new Map(String, int)
17: **for** t in terms **do**
18: **for** s in sentences **do**
19: **if** maxPairDistance(t.words()) > 3 **then**
20: nonCompact[t] += 1
21: **for** t in terms **do**
22: **if** nonCompact[t] > 1 \vee ($\exists t'$: t' in terms \wedge t in t' \wedge
 psupport[t] < 3) **then**
23: terms.remove(t)
24: adjs = new Set(String)
25: **for** s in sentences **do**
26: **if** $\exists t$: t in terms \wedge t in s **then**
27: adjs.add(POSTagger(s).getNearestAdj(t))
28: **for** s in sentences **do**
29: **if** $\neg \exists t$: t in terms \wedge t in s $\wedge \exists a$: a in adjs \wedge a in s
 then
30: t = POSTagger(s).getNearestNoun(adjs)
31: terms.add(t)
32: **return** psupport.keysSortedByValue()

4.3 The H&L+W2V method

We extended H&L by including an additional pruning step that uses continuous vector space representations of words (Mikolov et al., 2013a; Mikolov et al., 2013b; Mikolov et al., 2013c). The vector representations of the words are pro-

Centroid	Closest Wikipedia words
Com. lang.	only, however, so, way, because
Restaurants	meal, meals, breakfast, wingstreet, snacks
Hotels	restaurant, guests, residence, bed, hotels
Laptops	gameport, hardware, hd floppy, pcs, apple macintosh

Table 1: Wikipedia words closest to the common language and domain centroids.

duced by using a neural network language model, whose inputs are the vectors of the words occurring in each sentence, treated as latent variables to be learned. We used the English Wikipedia to train the language model and obtain word vectors, with 200 features per vector. Vectors for short phrases, in our case candidate multi-word aspect terms, are produced in a similar manner.¹¹

Our additional pruning stage is invoked immediately immediately after line 6 of Algorithm 1. It uses the ten most frequent candidate distinct aspect terms that are available up to that point (frequency taken to be the number of sentences that contain each candidate) and computes the centroid of their vectors, dubbed the *domain centroid*. Similarly, it computes the centroid of the 20 most frequent words of the Brown Corpus (news category), excluding stop-words and words shorter than three characters; this is the *common language centroid*. Any candidate distinct aspect term whose vector is closer to the common language centroid than the domain centroid is discarded, the intuition being that the candidate names a very general concept, rather than a domain-specific aspect.¹² We use cosine similarity to compute distances. Vectors obtained from Wikipedia are used in all cases.

To showcase the insight of our pruning step, Table 1 shows the five words from the English Wikipedia whose vectors are closest to the common language centroid and the three domain centroids. The words closest to the common language centroid are common words, whereas words closest to the domain centroids name domain-specific concepts that are more likely to be aspect terms.

¹¹We use WORD2VEC, available at <https://code.google.com/p/word2vec/>, with a continuous bag of words model, default parameters, the first billion characters of the English Wikipedia, and the pre-processing of <http://mattmahoney.net/dc/textdata.html>.

¹²WORD2VEC does not produce vectors for phrases longer than two words; thus, our pruning mechanism never discards candidate aspect terms of more than two words.

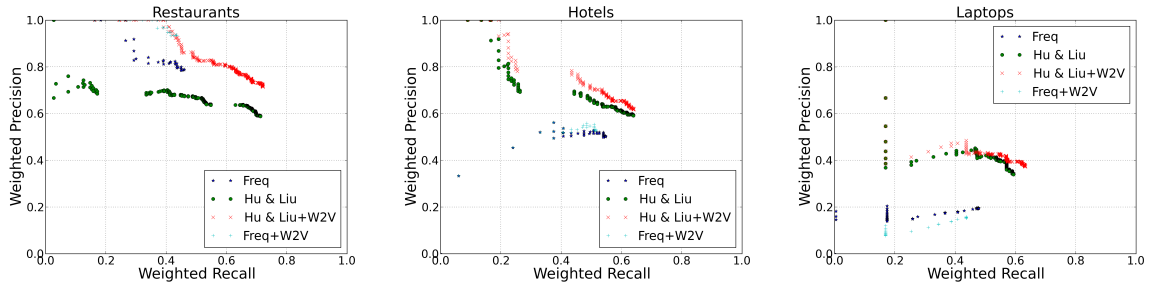


Figure 3: Weighted precision – weighted recall curves for the three datasets.

4.4 The FREQ+W2V method

As with H&L+W2V, we extended FREQ by adding our pruning step that uses the continuous space word (and phrase) vectors. Again, we produced one common language and three domain centroids, as before. Candidate distinct aspect terms whose vector was closer to the common language centroid than the domain centroid were discarded.

5 Experimental results

Table 2 shows the *AWP* scores of the methods. All four methods perform better on the restaurants dataset. At the other extreme, the laptops dataset seems to be the most difficult one; this is due to the fact that it contains many frequent nouns and noun phrases that are not aspect terms; it also contains more multi-word aspect terms (Fig. 2).

H&L performs much better than FREQ in all three domains, and our additional pruning (W2V) improves H&L in all three domains. By contrast FREQ benefits from W2V only in the restaurant reviews (but to a smaller degree than H&L), it benefits only marginally in the hotel reviews, and in the laptop reviews FREQ+W2V performs worse than FREQ. A possible explanation is that the list of candidate (distinct) aspect terms that FREQ produces already misses many aspect terms in the hotel and laptop datasets; hence, W2V, which can only prune aspect terms, cannot improve the results much, and in the case of laptops W2V has a negative effect, because it prunes several correct candidate aspect terms. All differences between *AWP* scores on the same dataset are statistically significant; we use stratified approximate randomization, which indicates $p \leq 0.01$ in all cases.¹³

Figure 3 shows the weighted precision and weighted recall curves of the four methods. In the restaurants dataset, our pruning improves

Method	Restaurants	Hotels	Laptops
FREQ	43.40	30.11	9.09
FREQ+W2V	45.17	30.54	7.18
H&L	52.23	49.73	34.34
H&L+W2V	66.80	53.37	38.93

Table 2: Average weighted precision results (%).

the weighted precision of both H&L and FREQ; by contrast it does not improve weighted recall, since it can only prune candidate aspect terms. The maximum weighted precision of FREQ+W2V is almost as good as that of H&L+W2V, but H&L+W2V (and H&L) reach much higher weighted recall scores. In the hotel reviews, W2V again improves the weighted precision of both H&L and FREQ, but to a smaller extent; again W2V does not improve weighted recall; also, H&L and H&L+W2V again reach higher weighted recall scores. In the laptop reviews, W2V marginally improves the weighted precision of H&L, but it lowers the weighted precision of FREQ; again H&L and H&L+W2V reach higher weighted recall scores. Overall, Fig. 3 confirms that H&L+W2V is the best method.

6 Conclusions

We constructed and made publicly available three new ATE datasets from three domains. We also introduced weighted variants of precision, recall, and average precision, arguing that they are more appropriate for ATE. Finally, we discussed how a popular unsupervised ATE method can be improved by adding a new pruning mechanism that uses continuous space vector representations of words and phrases. Using our datasets and evaluation measures, we showed that the improved method performs clearly better than the original one, also outperforming a simpler frequency-based baseline with or without our pruning.

¹³See <http://masanjin.net/sigtest.pdf>.

References

- A. Bagheri, M. Saracee, and F. Jong. 2013. An unsupervised aspect detection model for sentiment analysis of reviews. In *Proceedings of NLDB*, volume 7934, pages 140–151.
- J. Blitzer, M. Dredze, and F. Pereira. 2007. Biographies, Bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of ACL*, pages 440–447, Prague, Czech Republic.
- X. Ding, B. Liu, and P. S. Yu. 2008. A holistic lexicon-based approach to opinion mining. In *Proceedings of WSDM*, pages 231–240, Palo Alto, CA, USA.
- G. Ganu, N. Elhadad, and A. Marian. 2009. Beyond the stars: Improving rating predictions using review text content. In *Proceedings of WebDB*, Providence, RI, USA.
- M. Hu and B. Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of KDD*, pages 168–177, Seattle, WA, USA.
- Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems*, 20(4):422–446.
- S.-M. Kim and E. Hovy. 2006. Extracting opinions, opinion holders, and topics expressed in online news media text. In *Proceedings of SST*, pages 1–8, Sydney, Australia.
- N. Kobayashi, K. Inui, and Y. Matsumoto. 2007. Extracting aspect-evaluation and aspect-of relations in opinion mining. In *Proceedings of EMNLP-CoNLL*, pages 1065–1074, Prague, Czech Republic.
- B. Liu, M. Hu, and J. Cheng. 2005. Opinion observer: analyzing and comparing opinions on the web. In *Proceedings of WWW*, pages 342–351, Chiba, Japan.
- B. Liu. 2012. *Sentiment Analysis and Opinion Mining*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool.
- C. Long, J. Zhang, and X. Zhut. 2010. A review selection approach for accurate feature rating estimation. In *Proceedings of COLING*, pages 766–774, Beijing, China.
- C. D. Manning, P. Raghavan, and H. Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press.
- T. Mikolov, K. Chen, G. Corrado, and J. Dean. 2013a. Efficient estimation of word representations in vector space. In *Proceedings of Workshop at ICLR*.
- T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Proceedings of NIPS*.
- T. Mikolov, W.-T. Yih, and G. Zweig. 2013c. Linguistic regularities in continuous space word representations. In *Proceedings of NAACL HLT*.
- S. Moghaddam and M. Ester. 2010. Opinion digger: an unsupervised opinion miner from unstructured product reviews. In *Proceedings of CIKM*, pages 1825–1828, Toronto, ON, Canada.
- B. Pang and L. Lee. 2005. Seeing stars: exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of ACL*, pages 115–124, Ann Arbor, MI, USA.
- Ana-Maria Popescu and Oren Etzioni. 2005. Extracting product features and opinions from reviews. In *Proceedings of HLT-EMNLP*, pages 339–346, Vancouver, Canada.
- T. Sakai. 2004. Ranking the NTCIR systems based on multigrade relevance. In *Proceedings of AIRS*, pages 251–262, Beijing, China.
- B. Snyder and R. Barzilay. 2007. Multiple aspect ranking using the good grief algorithm. In *Proceedings of NAACL*, pages 300–307, Rochester, NY, USA.
- M. Tsytsarau and T. Palpanas. 2012. Survey on mining subjective data on the web. *Data Mining and Knowledge Discovery*, 24(3):478–514.
- C.-P. Wei, Y.-M. Chen, C.-S. Yang, and C. C. Yang. 2010. Understanding what concerns consumers: a semantic approach to product feature extraction from consumer reviews. *Information Systems and E-Business Management*, 8(2):149–167.
- J. Yu, Z. Zha, M. Wang, and T. Chua. 2011. Aspect ranking: identifying important product aspects from online consumer reviews. In *Proceedings of NAACL*, pages 1496–1505, Portland, OR, USA.
- Z. Zhai, B. Liu, H. Xu, and P. Jia. 2010. Grouping product features using semi-supervised learning with soft-constraints. In *Proceedings of COLING*, pages 1272–1280, Beijing, China.
- Z. Zhai, B. Liu, H. Xu, and P. Jia. 2011. Clustering product features for opinion mining. In *Proceedings of WSDM*, pages 347–354, Hong Kong, China.