Detection of Multiple Implicit Features per Sentence in Consumer Review Data

Nikoleta Dosoula, Roel Griep, Rick den Ridder, Rick Slangen,
Kim Schouten, and Flavius Frasincar

Erasmus University Rotterdam
PO Box 1738, NL-3000 DR
Rotterdam, the Netherlands
{384964nd,416133rg,324065rr,362941rs}@student.eur.nl,
{schouten,frasincar}@ese.eur.nl

Abstract. With the rise of e-commerce, online consumer reviews have become crucial for consumers' purchasing decisions. Most of the existing research focuses on the detection of explicit features and sentiments in such reviews, thereby ignoring all that is reviewed *implicitly*. This study builds, in extension of an existing implicit feature algorithm that can only assign one implicit feature to each sentence, a classifier that predicts the presence of multiple implicit features in sentences. The classifier makes its prediction based on a score function and is trained by means of a threshold. Only if this score exceeds the threshold, we allow for the detection of multiple implicit feature. In this way, we increase the recall while limiting the decrease in precision. In the more realistic scenario, the classifier-based approach improves the F_1-score by 1.6 percentage points on a restaurant review data set.

1 Introduction

In the last decade, a growing amount of retail activity is transferred from the street to the Web. Nowadays, people buy a wide range of consumer goods online using websites such as Amazon or Alibaba. These e-commerce companies often provide an easily accessible platform where consumers can share their experiences with and opinions about their purchases in the form of product reviews. As the required effort for writing these reviews becomes increasingly little, the number of product reviews on online retail shops sharply increased during the last decade. To illustrate this, in 2014 the number of reviews on Amazon exceeded the 10 million [5]. Furthermore, the number of online reviewing platforms, where consumers leave behind product or service reviews, continues to grow.

Using these product reviews for decision making has become increasingly popular [10]. Where some consumers might be looking for specific comments on their potential purchase, others might only be interested in the overall sentiment or in the sentiment per product aspect. However, the number of reviews can be high for some (popular) products, which makes reading all those reviews very time consuming. In order to lower these information costs, one of three pillars in

the classical transaction cost model [3], an automatic assessment of the overall sentiment within consumer reviews is asked for.

The main aim of this paper is to contribute to the existing research on the detection of implicit features within consumer reviews. In particular, we seek to extend the method proposed in [9] by adding a classifier that predicts the presence of multiple implicit features within a sentence. The evaluation of our method shows that we can significantly improve the F_1-measure by 1.6% compared to [9], resulting in an F_1-measure equal to 64.5%. Apart from increasing the F_1-measure, our method contributes to existing work by its suitability for a more realistic scenario in which sentences are allowed to have more than one implicit feature.

The remaining part of this paper is organized as follows. Sect. 2 reviews the relating literature and addresses the possible shortcomings of previously proposed methods. After presenting our method in Sect. 3, we discuss the data set used in our experiments in Sect. 4. Sect. 5 then discusses the implementation of our proposed method and we evaluate its performance in Sect.6, also by comparing it to previous work in the literature. Sect.7 concludes this paper and proposes possible avenues for future research.

2 Related Work

This section discusses the relevant literature in the field that is concerned with the automated assignment of implicit product features within consumer reviews. Our proposed method is motivated by the shortcomings of existing approaches.

The vast majority of approaches in the literature focuses on finding the explicit features in sentences. This limited approach is understandable because often in reviews most of the features are explicitly mentioned in a sentence. However, as addressed before, features that are implicitly mentioned in reviews are equally important. In feature-based sentiment analysis the detection of implicit feature therefore plays an essential role. However, in order to obtain reliable results, sophisticated methods that can infer implicit features from sentences are required. This section addresses some of the most relevant approaches.

A method of detecting implicit features is proposed in [8]. More specifically, the method refers to a two-phase co-occurrence association rule mining approach. In the first phase, [8] mines a set of association rules from co-occurrences between opinion words and explicit features. Therefore each opinion word is associated with a set of candidate features. In the second phase, the explicit features are clustered in order to obtain more powerful rules. If an opinion word is not linked with an explicit feature, the list of rules is checked in order to assign the most likely feature to this opinion word.

A similar approach to [8] is presented in [12]. Specifically, [12] mines as many association rules as possible between feature indicators and the corresponding features. Namely, the indicators are based on word segmentation, part-of-speech tagging, and feature clustering. As basic rules, the best rules in five different rule sets are chosen. In addition, three methods are proposed in [12] to find

some set of rules: adding substring rules, adding dependency rules, and adding constrained topic model rules. In the final stage, the results of both approaches are compared where the latter one, using expanding methods, shows the best performance.

One pioneering method for the detection of implicit features is the one of [14], which originates from the following basic idea. A set of several selected opinion words is constructed and the reviews are scanned for so-called modification relationships between these opinions words and corresponding explicit feature words within the same sentences. In other sentences, these opinion words could appear without the presence of an explicit feature. Based on the modification frequencies, a set of candidate features is then determined for these sentences. Then, a co-occurrence matrix is built in which the numbers of co-occurrences between all notional words, i.e. also between non-opinion words and features, are calculated. Using this co-occurrence matrix, constrained by the set of candidates features, the algorithm in [14] selects features using information from all notional words within a sentence. The candidate features that are chosen have co-occurred with the corresponding opinion word before. For example, in the case of digital camera reviews, if the word 'good' appears within the same sentence as the explicitly mentioned features 'battery', 'lens' and 'material', these would be candidate features for an opinion word 'good'. From this set of candidate set of features, the implicit feature is inferred according to the associations between these candidate feature words and the rest of the notional words in the sentence, which are stored in the co-occurrence matrix.

It is important to keep the above described method in mind, since it forms an important building block of the method used by [9], on which this paper expands. The main difference in the approach by [9] is that it uses a supervised algorithm. Namely, consumers review data is used in which all implicit features are annotated. Therefore, co-occurrences can be calculated between these annotated implicit features and all words in the sentence. Based on the co-occurrences, scores are then assigned to potential implicit features, which in the case of [9] are *all* implicit features within the data set. Finally, the implicit feature with the highest score is assigned to the sentence. An advantage of [9], is that it can also be used to detect features that are not present explicitly within the data set. This is an improvement over the methods presented in [8], [12], and [14], where implicit features can only be detected when they also appear explicitly in the data set. Nevertheless, it relies on the existence of training data that is annotated with implicit features.

Furthermore, [9] improves on [14] by introducing a trained threshold in the assignment of implicit features. Where in the method presented in [14] relative low co-occurrence scores could already lead to linking an opinion word to a feature, the algorithm in [9] only assigns an implicit feature to a sentence when its score exceeds the learned threshold. The idea behind this is that when the co-occurrence frequencies are low, it is questionable whether the sentence should be linked to any feature at all. Especially in the case when there are many sentences

without any implicit feature, the improvements by using such a threshold show to be large [9].

However, one apparent disadvantage of the detection procedure by [9] and [14] is that it rules out the possibility that a sentence contains two or more implicit features. This seems an unrealistic constraint, especially in the field of product reviews, where people are explicitly asked for their opinion. In fact, sentences containing two or more implicit features appear quite frequently. For instance, [4] makes the following observation in tweets that were collected from Twitter for their sentiment analysis: even short sentences may contain multiple sentiment types, concerning possibly different topics, e.g. *#fun* and *#scary* in "*Oh My God http://goo.gl/fb/K2N5z #entertainment #fun #pictures #photography #scary #teaparty*". [15] sees the same tendency in product review data. From their Chinese restaurant review data, an intuitive example is extracted. In the sentence "the fish is great, but the food is very expensive", two obvious sentiment words can be noticed: 'great' and 'expensive'. Both these words implicitly refer to two different features which could be labeled respectively as 'quality' and 'price'.

3 Method

This section discusses our method that works as an extension on the algorithm developed by [9] in the sense that it allows for the extraction of multiple features per sentence. This more unrestrictive approach considers a more realistic scenario, in which sentences can be related to multiple implicit features.

We start with a short, formal description of the algorithm earlier presented in [9]. From the training data, the algorithm stores all unique annotated implicit features and all unique lemmas (which are the syntactic root form of a word) with their frequencies in list F and O. Furthermore, $|F| \times |O|$ matrix C stores the co-occurrences between all elements in F and O within sentences. Then, sentences in the test data are processed as follows. For each ith implicit feature $f_i \in F$, the sum of the ratios between the co-occurrence $c_{i,j} \in C$ of each jth word in the sentence and the frequency $o_j \in O$ of that word is calculated:

$$Score_{f_i} = \frac{1}{n} \sum_{j=1}^{n} \frac{c_{i,j}}{o_j}, \tag{1}$$

where n is the number of words in a sentence. Finally, the implicit feature with the highest score is assigned to the sentence when it exceeds a trained threshold. When there is no score that exceeds the threshold, no feature is assigned to the sentence. The training of the threshold is only based on the training data and is executed by simply finding the threshold value between 0 and 1 which yields the best performance.

One approach to extend the algorithm to a more realistic scenario is by selecting all implicit features that exceed the trained threshold (see Sect. 2). However, when only a small proportion of the data set consists of sentences that contain more than one implicit features, the precision of the algorithm would

Algorithm 1 Algorithm training using annotated data.

```
Construct list F of unique implicit features
Construct list O of unique lemmas with frequencies
Construct co-occurence matrix C
for all sentence s ∈ training data do
  for all word w ∈ s do
    if ¬(w ∈ O) then
      add w to O
    end if
    O(w) = O(w) + 1
  end for
  for all implicit feature f ∈ s do
    if ¬(f ∈ F) then
      add f to F
    end if
    for all word w ∈ s do
      if ¬((w, f) ∈ C) then
        add (w, f) to C
      end if
      C(w, f) = C(w, f) + 1
    end for
  end for
end for
Train threshold for the classifier through linear search
Train threshold for the feature detection algorithm through linear search
```

suffer from such a crude selection mechanism. To understand this effect, one should realize that when specific words co-occur often with different implicit features, sentences in which these words are present consequently have a high score for more than one implicit feature. However, assigning more than one implicit feature to each of such sentences based on these scores might be naive when only few sentences are known to contain more than one implicit feature. Another approach to allow for multiple features is to use a classifier to determine the number of implicit features that is likely to be present within the sentence. Subsequently, the algorithm could assign features with top scores to a sentence, where now the number of assignments is based on the classifier's prediction. One should bear in mind however that this strategy now potentially suffers from the imperfect nature of both the classifier and the implicit feature extraction algorithm, which possibly leads to lower precision.

The method that we present works as a combination of the two above-mentioned methods such that we can utilize the advantages of both while minimizing their disadvantages. In particular, we use a classifier in order to detect for every sentence whether there it contains more than one implicit features. If the classifier predicts more than one implicit feature, all features with a score exceeding the threshold will be assigned to the sentence. Otherwise, only the feature with the highest score could be assigned to the sentence, that is, if it

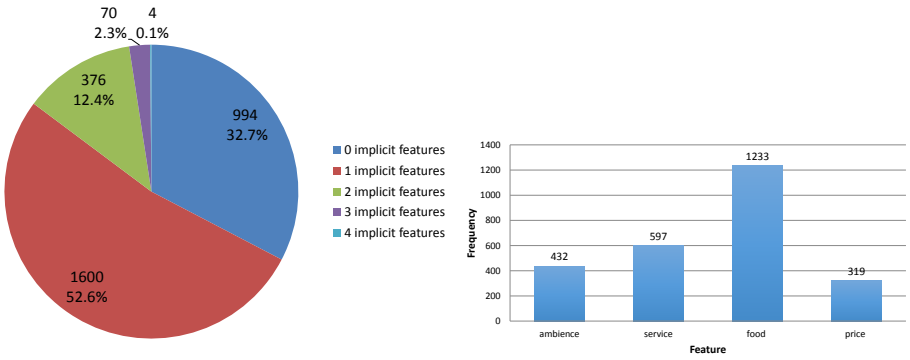
Algorithm 2 Algorithm execution on new sentences in the test data.

Input: trained thresholds $kThreshold$ and $fThreshold$
Construct list NN with the number of nouns per sentence
Construct list JJ with the number of adjectives per sentence
Construct list CM with the number of commas per sentence
Construct list A with the number of 'and' words per sentence
Obtain $\hat{\beta}$'s from logistic regression using the full data set
for all sentence $s \in$ test data **do**
  $kScore = \hat{\beta}_0 + \hat{\beta}_1 NN(s) + \hat{\beta}_2 JJ(s) + \hat{\beta}_3 CM(s) + \hat{\beta}_4 A(s)$
  $currentBestFeature = empty$
  $fScoreOfCurrentBestFeature = 0$
  for all feature $f \in F$ **do**
    $fScore = 0$
    for all word $w \in s$ **do**
      $fScore = fScore + C(w, f)/O(w)$
    end for
    if $kScore > kThreshold$ **then**
      if $fScore > fThreshold$ **then**
        Assign feature f to s
      end if
    else if $fScore > fScoreOfCurrentBestFeature$ **then**
      $currentBestFeature = f$
      $fScoreOfCurrentBestFeature = fScore$
    end if
  end for
  if $\neg(kScore > kThreshold)$ **then**
    if $fScoreOfCurrentBestFeature > fThreshold$ **then**
      Assign $currentBestFeature$ to s
    end if
  end if
end for

exceeds the trained threshold. Hence, the classifier produces the binary result whether or not to allow for multiple features. The pseudocode describing the described method is shown in Alg. 1 and Alg. 2.

The classifier calculates a score based on a number of sentence characteristics that are related with the number of implicit features k_s within a sentence s. When the score for a sentence exceeds another trained threshold, the classifier predicts multiple implicit features to be present. The score function uses the following variables: (i) number of nouns ($\texttt{\#NN}_s$), (ii) number of adjectives ($\texttt{\#JJ}_s$), (iii) number of commas ($\texttt{\#Comma}_s$), and (iv) the number of 'and' words ($\texttt{\#And}_s$). In order to determine the relation between these predictor variables and the number of implicit features, we estimate the following logistic regression equation by maximum-likelihood:

$$Score_{k_s} = \log\left(\frac{p_s}{1-p_s}\right) = \beta_0 + \beta_1 \texttt{\#NN}_s + \beta_2 \texttt{\#JJ}_s + \beta_3 \texttt{\#Comma}_s + \beta_4 \texttt{\#And}_s, \quad (2)$$



(a) Distribution of the number of implicit features contained per sentence, in the restaurant review data set.

(b) Frequencies of the four unique implicit features in our data set.

where p_s is the probability that sentence s contains multiple implicit features. The coefficients are estimated using the full data set. The implementation of this regression approach is discussed in more detail in Sect. 5.

This extended algorithm is now trained in two steps, only using the training data. First, the threshold for the classifier is trained in terms of prediction performance. Second, the threshold for the feature detection algorithm, now using the prediction of the classifier, is trained (as described in the second paragraph of this section) to optimize the feature detection performance.

As a final remark, a limitation of this method is that it requires a sufficiently large data set in which the implicit features are annotated. The reason for this is that the training of the algorithm is executed on annotated implicit features. However, the benefit of this approach is that the algorithm is now able to detect all implicit features within the data set, and not only the features that are (also) *explicitly* present in the data set.

4 Data Analysis

The data set which is used to build up and validate the method proposed in the previous section consists of a collection of restaurants reviews [7]. Every review sentence is assigned to at least one of five so-called review aspect categories: 'food', 'service', 'ambience', 'price', and 'anecdotes/miscellaneous'. These aspect categories are generally not explicitly referred to in a sentence but can be inferred from each sentence. Therefore, these aspect categories operate as *implicit* features of the product, i.e., the restaurant. In the data set, both implicit and explicit restaurant features are labeled.

All 3,044 sentences in the restaurant data set contain at least one implicit feature. However, in order to obtain a better performance test of our classifier for the number of implicit features present in each sentence, the fifth category of

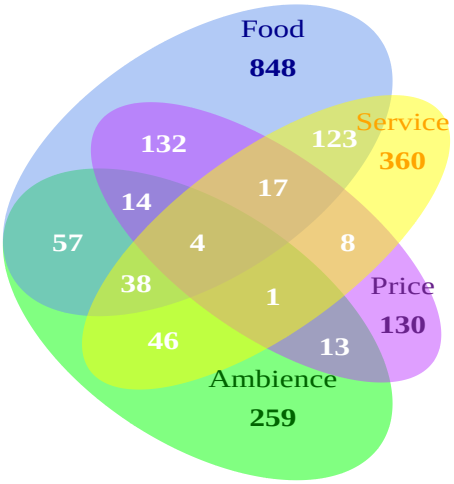


Fig. 2. Co-occurrence frequencies of the four unique implicit features in our data set.

'anecdotes/miscellaneous' is removed from the data set. This particular category seems most appropriate for removal, as it does not describe a unique implicit feature but refers to the general context 'miscellaneous'. In this way, the number of implicit features in our data set has a wider distribution because part of the set now consists of sentences without an implicit feature. As consumer review sentences generally do not always contain an implicit feature, the performance of our classifier on this more realistic scenario is interesting. Furthermore, in this setting the influence of the threshold parameter in the algorithm by [9] in combination with our classifier can be measured.

As clearly displayed in Fig. 1a, more than half of the sentences contain only one implicit feature. However, in a significant percentage of sentences, namely 12.4%, two implicit features are mentioned. This motivates an approach that considers more than one implicit feature in a sentence.

Examining the frequency of the four implicit features in Fig. 1b, it is clear that all of them play an important role in customer's reviews. Interesting is the fact that each of them appears in more than 300 sentences which is because there is only a small set of features. More specifically, 'food' captures more than one third of the sentences in total and more than twice of any of the other categories. In terms of frequency, 'food' is followed by the feature 'service' appearing in nearly half as many sentences as 'food'. Feature 'ambience' is implicitly referred to in 432 sentences. Lastly, the least common feature is 'price', where the difference with 'food' is a factor of three.

As the main purpose of the method that we propose is to search for multiple implicit features in each sentence, it seems worthwhile to examine to what extent multiple features are present in one sentence. Fig. 2 shows the frequency of all possible co-occurrences between the four unique implicit features. Clearly, most

Table 1. Coefficients of logistic regression (2) for the classifier.

Predictor Variable	Coefficient	p-value
`Constant`	−3.019479	0.0000
$\texttt{\#NN}_s$	0.116899	0.0002
$\texttt{\#JJ}_s$	0.335530	0.0000
$\texttt{Comma}_s$	0.216417	0.0004
$\texttt{And}_s$	0.399415	0.0000

of the sentences in our data contain only one implicit feature, something that can also be seen in Fig. 1a. More than 4% of the sentences implicitly refer to both 'food' and 'price', and almost the same percentage corresponds to the co-occurrence of 'food' and 'service'. The remaining combinations of two implicit features appear less frequently in the same sentence in our restaurant review data set.

5 Implementation

To predict the presence of multiple implicit features, we use the score function as given in Eq. 2. We think of this score function as a general rule for categorized review data such as our restaurant review data set. In order to specify the correct score function, however, sufficient amount of this type of consumer review data is required. Constrained by resources, however, only the same restaurant review data set is available to us. Therefore, the score function is not trained on a training part of the data set, and then tested on a test part. Instead, the full data set is used in order to maximize the information available to us.

We estimate the $Score_{k_s}$ function (2) using logistic regression. Table 1 displays the results. The p-values indicate that our variables are highly significant, i.e., for significance levels below 1%. Apart from the variables that we include, we also test implementing the number of words in a sentence and the number of grammatical subjects in a sentence. Neither of these variables yield a significant improvement. Intuitively, this can be explained because the variables for the number of nouns and adjectives already capture the relevant information that lies within the number of words within a sentence. The number of subjects possibly does not perform better than the number of nouns because often the subject in a sentence is the product instead of the feature.

The regression is performed on the complete restaurant data set, as motivated above. However, one could argue that this could result in unfair performance, as the same data set is used to evaluate our algorithm. However, when the coefficients of the regression are robust for different subsamples, specifying a different score function based on an arbitrary train part of the data set will not alter the results heavily. Put differently, this would indicate that our approach of using the full data set does not provide an unfair edge. To check whether this is

Table 2. Specifications of 1000 logistic regressions on 90% subsamples.

Variable	Mean	Median	Std. dev.
$\texttt{\#NN}_s$	0.117361	0.11768	0.011342
$\texttt{\#JJ}_s$	0.335538	0.33536	0.014345
$\texttt{Comma}_s$	0.216409	0.21672	0.023185
$\texttt{And}_s$	0.399507	0.39892	0.023409

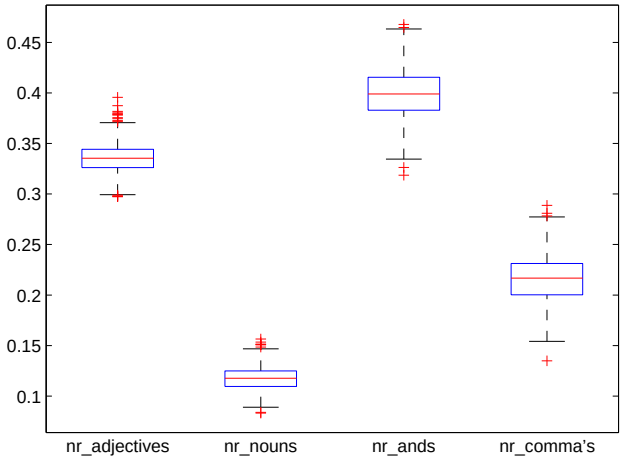


Fig. 3. Box-plot of the coefficients of the logistic regression.

the case, we perform the logistic regression 1000 times on arbitrary subsamples containing 90% of the data set. Fig. 3 depicts the coefficients of the 1000 regressions in a box-plot and Table 2 provides descriptive statistics. The constant is excluded from the plot and table, because it does not influence the result with a trained threshold. We find that the values of the coefficients do not differ a lot for the different subsamples, so it is justified to use the complete data set when determining the coefficients.

The classifier predicts multiple implicit features for sentence s when Score_{k_s} is larger than a certain threshold. We can therefore train the classifier by determining the optimal threshold. In order to do so, we isolate the performance of the classifier by assuming that the feature detection part of the algorithm is perfect. That is, if the classifier predicts the presence of multiple implicit features correctly, we assign all golden implicit features to that sentence; if the classifier predicts incorrectly, we assign either only one golden implicit feature (in case there are actually multiple implicit features), or one implicit feature too many (in case there are not actually multiple features) to the sentence. This way, the

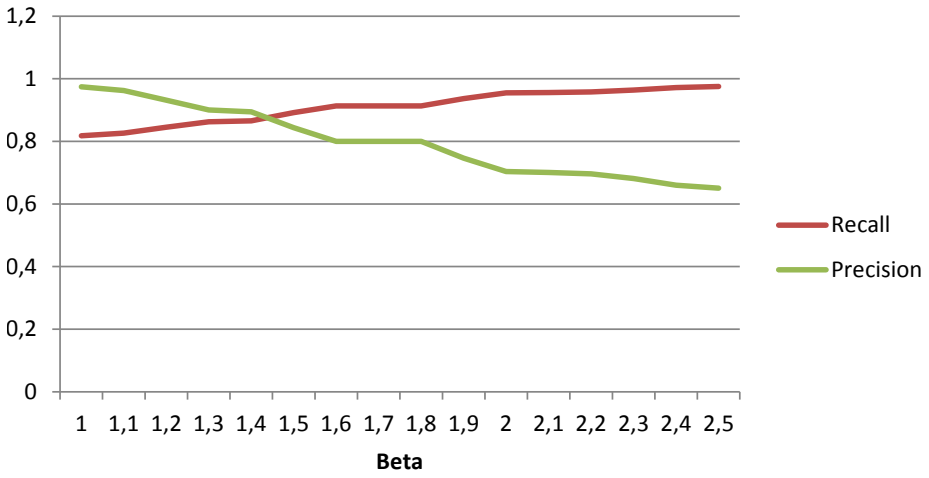


Fig. 4. Recall and precision of the classifier for different β's.

errors made by the classifier are isolated and can thus be minimized by means of altering the threshold.

The classifier is optimized on F_β-score. Since the main goal of our classifier is to predict multiple implicit features when they are present, high recall is especially important. If it incorrectly predicts no multiple implicit features in the sentence, the recall of the final score will always decrease, because there can only be one implicit feature assigned to that sentence. However, when the classifier incorrectly predicts the presence of multiple implicit features, the precision of the final score does not necessarily decrease. The threshold in the feature detection part of the algorithm could prevent that multiple implicit features are assigned to a sentence. Fig. 4 shows the precision and recall of the classifier with a trained threshold for different β's. It can be seen that with β larger than 1.8, the precision decreases relatively fast, while the recall only increases a little bit. Therefore, we use β equal to 1.8 in the F_β-score when training the classifier to emphasize recall.

Finally, the threshold is trained on an annotated training set containing 90% of the data. To train the threshold, a range of threshold values needs to be defined. We use values between -3 and 3, with a step size of 0.1. With every threshold, the classifier is evaluated based on $F_{1.8}$. Hence, after linearly trying all possible thresholds, we use the threshold with the largest $\mathrm{F}_{1.8}$-score.

6 Evaluation

Evaluation of the implemented method is based on 10-fold cross-evaluation. This means that the whole data set is split into two subsets: one part contains 90% of the data, the other part 10%. The algorithm is then trained on this 90% of the data set. The trained algorithm then detects the implicit features in the remaining 10% of the data. This procedure is repeated 10 times, where there is

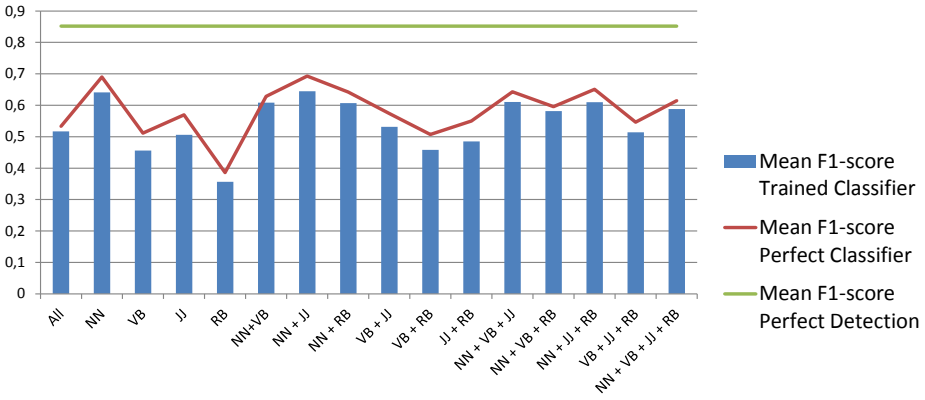


Fig. 5. Mean F_1-scores with different part-of-speech filters.

no overlap in the 10 hold-out samples. For each fold, the F_1-score is calculated and finally averaged to provide the measure for the performance of the algorithm.

The predictive performance we consider to evaluate the predicting of implicit features is the F_1-score . Using the F_1-score as the performance measure allows for easy comparison with previous work, as it is one of the standard performance measures within the literature.

Because the different training and test subsamples used in the cross-evaluation are generated randomly, we run our algorithm 10 times. Fig. 5 shows the results, in terms of mean F_1-scores, following from our proposed method (the blue bars). To provide more insights into our results, Fig. 5 also depicts F_1-scores of the algorithm with both a perfect classifier (the red line) and with a perfect feature detection algorithm (the green line). The scores using a perfect classifier are computed by always passing the correct prediction (in terms of the presence of multiple implicit features) onto the feature detection algorithm. The scores with the perfect feature detection are found by, based on the prediction of the classifier, assigning a number of golden implicit features to the sentences.

Results are given for different part-of-speech filters, which are used to filter out possibly irrelevant words in the co-occurrence matrix that could be harmful to the performance of the algorithm. Fig. 5 shows the scores for 16 different part-of-speech filters. The filters include only the words of types that are mentioned, where `NN` stands for nouns, `VB` for verbs, `JJ`, for adjectives and `RB` for adverbs. Examining the F_1-scores in Fig. 5, we find that the best results are obtained using the `NN+JJ` part-of-speech filter. That is, filtering for nouns and adjectives, we obtain an F_1-score equal to 64.5%. We note that the F_1-score we find for using the `NN` filter is only marginally worse, namely 64.1%.

Since our proposed method extends the one presented in [9], we start by evaluating the increase in performance as a result of our extension. In order to do so, we also evaluate the unextended algorithm as presented in [9] 10 times using the `NN+JJ` part-of-speech filter. Again, we note that each evaluation provides slightly

different results due to the random nature of the cross-evaluation method. We find a mean F_1-score of 62.9% for the algorithm without the classifier.[1] Hence, comparing this to our 64.5%, we find an improvement of 1.6 percentage points. We test for significance by means of a two-sample t-test. This results in a t-test statistic equal to 12.0, which indicates a significant improvement at the significance level of 1%.

At first sight, an improvement in mean F_1-score of 1.6 percentage points may not look large. However, in order to make a fair statement about the performance of our classifier, we first need to compare it to its potential, which is displayed by the red line in Fig. 5. We see that the potential is also at its largest for the `NN+JJ` filter, giving an F_1-score of 69.3%. This means that our classifier captures 25% of the maximum improvement that can be gained by adding a classifier that predicts the presence of multiple implicit features within a sentence, which is 6.4%. However, we note that the potential of such a classifier depends on the data set. In our restaurant review data set, 14.8% of the sentences contain more than one implicit feature, where 12.4% of the sentences contain two implicit features. Furthermore, calculations show that 20.4% of the total possible implicit features remain to be detected when only one implicit feature per sentence is considered.[2] However, it is important to notice that the most apparent implicit feature in each sentence is already detected. As a result, the second implicit feature would be assigned with an already lower precision, tempering the improvement of the F_1-score due to a higher recall. Therefore, in light of these insights and considering the simplicity of our approach, we consider our gained improvement to be significant.

Lastly, we provide insights in our results by looking at the F_1-score that can be obtained by using our classifier in combination with a perfect feature detection algorithm, which is displayed by the green line in Fig. 5. Notice that our classifier is trained to maximize the $F_{1.8}$-score. For this reason, these 'potential' F_1-scores are hardly interpretable and a greater potential might be visible when the classifier is trained for the same measure by which it is now evaluated. However, training for $F_{1.8}$ yields best *overall* performance in our method, which is also motivated in Sect. 5. Nonetheless, these F_1-scores provide insight in what part in the loss of F_1-score can be attributed to the feature detection part of our algorithm. The F_1-scores with perfect detection, which do not rely on the part-of-speech filters, are 85.2%. Comparing this result with the one in the previous paragraph, we conclude that improving the feature detection part of the algorithm shows greater potential than improving the prediction of the presence of multiple implicit features.

[1] We note that in [9], based on a number of runs, a maximum F_1-score of 63.3% is reported.

[2] Based on the distribution of the number of implicit features per sentence in our data set (see Fig. 1a), we have: $(12.4+2\cdot2.3+3\cdot0.1)/(52.6+2\cdot12.4+3\cdot2.3+4\cdot0.1) = 0.204$.

7 Conclusion

In many of the existing methods within the literature, detection algorithms are limited to assigning only one implicit feature per sentence. However, when consumers review their purchased products, they do typically not obey this constraint. Therefore, based on this visible shortcoming in previous work, we propose an algorithm that allows for the detection of multiple implicit features per sentence. Our method directly extends the more constrained, supervised method earlier proposed in [9].

In our proposed method we construct a classifier that predicts the presence of multiple implicit features using a score function. The score function is based on four simple sentence characteristics: (i) number of nouns, (ii) number of adjectives, (iii) number of commas, and (iv) the number of 'and' words. The function parameters are estimated by means of logistic regression and we train a threshold for better performance. Based on the prediction of the classifier for a given review, the feature detection part of our algorithm then looks for either one or multiple implicit features.

Considered on a restaurant review data set, our approach shows small but significant improvement with respect to the constrained method in [9]. That is, we improve the F_1-measure by 1.6 percentage points. Based on analysis of the performance of our classifier we conclude that we capture a reasonable (considering its simplicity) part of the full potential of our approach. The performance and potential of the classifier is however dependent on the distribution of the number of implicit features per sentence within the data set. That is, when consumer reviews frequently cover multiple implicit features per sentence, our more realistic approach is desirable.

In our approach we determine a *general* relation between sentences written in consumer reviews and the number of implicit features. Nonetheless, it might be desirable to integrate the specification and estimation of this relation in the training part of the algorithm in order to make it specifically effective for a given data set. One promising path for future work is therefore to train a classifier for the number of implicit features by using more advanced machine-learning techniques, such as Support Vector Machines. Also, rule learning methods could be employed in order to determine more indicators for the presence of multiple implicit features.

Another interesting suggestion for future research may be to combine the classifier with sentiment analysis algorithms. Namely, when there are opposing sentiments within one sentence, it seems likely that the consumer is commenting on two different features of the product. To illustrate this idea, we provide the following example:

"The phone looks great, but the pictures it takes are of very low quality."

In this sentence, two features are implied: 'appearance' and 'camera'. Also, there are two sentiment polarities: the consumer is positive about the appearance, but negative about the camera.

Acknowledgments

The authors are partially supported by the Dutch national program COMMIT.

References

1. Choi, F.: Advances in domain independent linear text segmentation. In: Proceedings of the first conference on North American chapter of the Association for Computational Linguistics. pp. 26–33 (2000)
2. Church, K.: Char align: A program for aligning parallel texts at the character level. In: Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics. pp. 1–8. Association for Computational Linguistics (1993)
3. Dahlman, C.: The problem of externality. Journal of Law and Economics 22(1), 141–162 (1979)
4. Davidov, D., Tsur, O., Rappoport, A.: Enhanced sentiment learning using twitter hashtags and smileys. In: COLING Proceedings of the 23rd International Conference on Computational Linguistics. pp. 241–249 (2010)
5. Floyd, K., Freling, R., Alhoqail, S., Cho, H., Freling, T.: How online product reviews affect retail sales: A meta-analysis. Journal of Retailing 90(2), 217–232 (2014)
6. Fragkou, P., Petridis, V., Kehagias, A.: A dynamic programming algorithm for linear text segmentation. Journal of Intelligent Information Systems 23(2), 179–197 (2004)
7. Ganu, G., Elhadad, N., Marian, A.: Beyond the stars: Improving rating predictions using review content. In: Proceedings of the 12th International Workshop on the Web and Databases (WebDB 2009). Rhode Island, USA (2009)
8. Hai, Z., Chang, K., Kim, J.: Implicit feature identification via co-occurrence association rule mining. In: Springer (ed.) Proceedings of the 12th International Conference on Computational Linguistics and Intelligent Text processing (CICLing 2011). vol. 6608, pp. 393–404 (2011)
9. Schouten, K., Frasincar, F.: Finding implicit features in consumer reviews for sentiment analysis. In: 14th International Conference on Web Engineering (ICWE 2014), volume 8541 of Lecture Notes in Computer Science. pp. 130–144. International World Wide Web Conferences Steering Committee (2014)
10. Senecal, S., Nantel, J.: The Influence of Online Product Recommendations on Consumers' Online Choices. Journal of Retailing 80, 159–169 (2004)
11. Van Rijsbergen, C.: Information Retrieval. Butterworth, London, 2nd edn. (1979)
12. Wang, W., Xu, H., Wan, W.: Implicit feature identification via hybrid association rule mining. Expert Systems with Applications 40(9), 3518–3531 (2013)
13. Zhang, F., Zhang, Z., Lan, M.: Ecnu: A combination method and multiple features for aspect extraction and sentiment polarity classification. In: Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014). pp. 252–258,. Dublin, Ireland (2014)
14. Zhang, Y., Zhu, W.: Extracting implicit features in online customer reviews for opinion mining. In: Proceedings of the 22nd International Conference on World Wide Web Companion (WWW 2013 Companion). pp. 103–104. International World Wide Web Conferences Steering Committee (2013)
15. Zhu, J., Wang, H., Zhu, M., Tsou, B., Ma, M.: Aspect-based opinion polling from customer reviews. IEEE Transactions On Affective Computing 2, 37–49 (2011)