Learning and Knowledge-Based Sentiment Analysis in Movie Review Key Excerpts

Björn Schuller[1] and Tobias Knaup[2]

[1] Institute for Human-Machine Communication
Technische Universität München, Germany
[2] Pingsta Inc., 1700 Seaport Boulevard, Redwood City, CA 94063, USA
{schuller}@tum.de

1 Introduction

Sentiment analysis has been studied for a number of different application domains: Among the most popular ones are product reviews [24,3,29,19,12,4], the stock market [2], hotels and travel destinations [24,19], and movie reviews [24,17,31]. While good results have been reported for product reviews, movie reviews seem to be more difficult to handle. In [24], a 66 % accuracy of valence estimation is reported for movie reviews, while up to 84 % are obtained for automobile reviews with the same method. The author explains this by a discrepancy between the semantic orientation of words that describe the elements of a movie (i. e., a scene, the plot), and its style or art. Obviously, movie reviews are exceptionally challenging for sentiment analysis. Motivated by these facts, the main goals of this chapter are: to introduce a novel approach to valence estimation that is based on on-line knowledge sources and linguistic methods, incorporate on-line knowledge sources into a machine learning approach, introduce an annotated database of over 100 k movie reviews collected from the review website Metacritic[1]—the largest to-date, experiment with both machine learning and

[1] http://www.metacritic.com

linguistic methods for the first time on a movie review database of that size, and—additionally—use Metacritic's fine-grained review scores as ground truth for a regression approach that leads to a higher resolution of valence prediction.

This remainder of this chapter is organized as follows: Section 2 introduces the Metacritic movie review corpus for analysis. Popular on-line knowledge sources are introduced in section 3. The on-line knowledge sources-based, syntax-driven algorithm we developed is described in section 4. It is evaluated in extensive experiments against data-driven modeling in section 5, and concluding remarks are given in section 6.

2 Metacritic – A Movie Review Database

To the best knowledge of the authors, the database that is described in this section is the largest movie review corpus to date. Several such corpora have been presented in the past, ranging in size from 120 to 2 000 reviews. Turney [24] collected 120 movie reviews from Epinions[2]. Zhuang et al. [31] selected 11 movies from the top 250 list of IMDB[3], and retrieved the first 100 reviews for each one of them. The data set contains a total of 16 k sentences and 260 k words. Probably the most widely used movie review database was introduced in [17]. In its initial version, 752 negative, and 1 301 positive reviews from the usenet newsgroup *rec.arts.movies.reviews* were used. Different versions have been released on the author's website[4], the current one containing 1 000 negative and 1 000 positive reviews.

Compared to these language resources, our Metacritic movie review corpus is significantly larger, containing a total of 102 622 reviews for 4 901 movies. Metacritic[5] is a website that compiles reviews for films, video/DVDs, books, music, television series, and computer games from various sources. An automated crawler was used to retrieve the HTML source of the webpages and store it for further processing. Because this chapter focuses on movies, only reviews from the film and video/DVD sections are included in the corpus that is described here. The web pages have been retrieved on January 8, 2009 between 12 p.m. and 2 p.m. CET. Reviews in Metacritic are available as excerpts of key sentences taken from the original text. The accumulated number of sentences is 133 394, and the total number of words in the database is 2 482 605. On average, a review has a length of 1.3 sentences, with a standard deviation of 0.6. Thus, Metacritic contains mostly short statements rather than long texts. The average length in words is 24.2, ranging between 1 and 104, with a standard deviation of 12.4 (cf. table 1). The vocabulary of the database has a size of 83 328 words. Looking at the breakdown of word classes, nouns (683 259) are the most frequent, followed by verbs (382 822), adjectives (244 825), and adverbs (174 152).

[2] http://www.epinions.com

[3] http://www.imdb.com

[4] http://www.cs.cornell.edu/people/pabo/movie-review-data/

[5] http://www.metacritic.com

Table 1. Size of Metacritic film and movie reviews. The minimum, maximum, average, and standard deviation (Std. Dev.) are given for each measure.

#	Minimum	Maximum	Average	Std. Dev.
Reviews per Movie	1	65	21.1	10.3
Word Count	1	104	24.2	12.4
Sentence Count	1	13	1.3	0.6

What makes Metacritic especially valuable is the fine-grained score value that accompanies every review. It is an integer score value ranging from 0 to 100, with higher scores indicating a better review. Metacritic scores are calculated from the original numeric rating scheme used by each source. Since the score is in most cases decided on by the author of the review, it can be considered very accurate. If no score is available from the author, though, it is assigned by a Metacritic staffer. Metacritic has its own classification schema that maps scores into three valence classes: positive/mixed/negative. Depending on the subject type, different mappings are used. The mapping for movies is shown in table 2.

Table 2. Mapping of score to valence class in Metacritic. Numbers apply to movies.

Score	Valence Class	# Reviews
81 – 100	positive	15 353
61 – 80	positive	38 766
40 – 60	mixed	32 586
20 – 39	negative	13 194
0 – 19	negative	2 723

This score and class label, and the huge number of reviews make Metacritic an ideal corpus for sentiment analysis: The accuracy of machine learning algorithms depends on the amount of available training data. Usually, a considerable amount of manual annotation work is required to build a corpus that is large enough to yield satisfying results. However, since the Metacritic score and class label can be directly used as ground truth for regression and classification respectively, no further annotation is needed.

The number of reviews in each class differs largely, as can be seen in table 3: There are about three times more positive reviews than negative reviews. Since machine learning techniques require a subset of the database to be used for training purposes, we chose to split the data into one set containing only reviews that have been released in odd years, and another set covering the even years. The resulting review subsets are of almost equal size: There are 49 698 instances in the ‘odd’ set, and 52 924 instances in the ‘even’ set.

The histogram of scores in figure 1 clearly shows that scores are not evenly distributed within each of the classes. Instead there are spikes for every score S that is a multiple of ten. For $S > 60$ (positive), there are also many bars

Table 3. Number and percentage of reviews in Metacritic falling into all, odd, and even years of release

# Reviews	All years	Odd years	Even years
Mixed	32 586 (31.75 %)	15 756 (31.70 %)	16 830 (31.80 %)
Positive	54 119 (52.74 %)	26 410 (53.14 %)	27 709 (52.36 %)
Negative	15 917 (15.51 %)	7 532 (15.16 %)	8 385 (15.84 %)

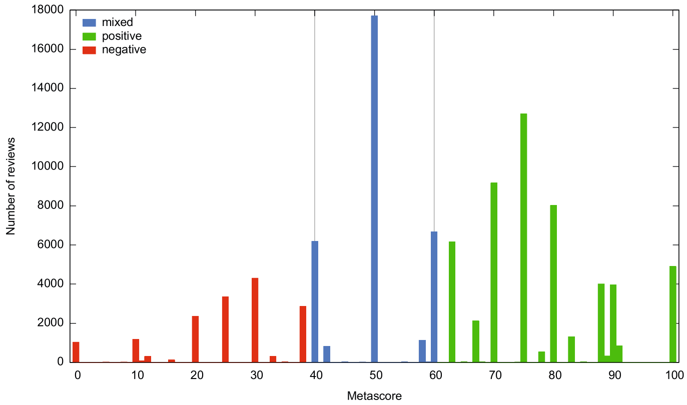

Fig. 1. Histogram of scores in the Metacritic database

in between, but especially for $40 \leq S \leq 60$ (mixed) there are almost only the bars at 40, 50, and 60. This is due to the fact that scores are converted from the original numerical rating schemas, and not all of these are as fine-grained as Metacritic, resulting in only a few values on the 0 – 100 scale.

3 On-line Knowledge Sources

On-line knowledge sources in natural language processing are databases of linguistic knowledge that are publicly available on the Internet. They contain information about words, concepts, or phrases, as well as connections among them. The kind of connection can be of different nature, such as commonsense knowledge, or lexical relations. Most databases focus on one of these fields. Different representational schemes have been developed to provide the information in a suitable and efficient way. In semantic networks, words or concepts are represented as nodes in a graph, and relations are represented by named links [10]. Another form are annotated dictionaries, where properties of a term are stored as tags. Note that dictionaries usually do not contain relations between terms. In the rest of this section, we will introduce some well-known on-line knowledge sources that are used in this chapter.

ConceptNet is a semantic network which contains commonsense knowledge in a machine-readable format. The knowledge is added by non-specialized humans through a data acquisition interface[6]. A lot of effort has been put into making

[6] http://commons.media.mit.edu/en/

the interface capable of ruling out false claims and other mistakes [8]. As the name implies, ConceptNet is a network of concepts, such as *"actor"* or *"to watch a movie"*. The storage format does not contain syntactic category information and therefore has no support for word sense disambiguation. But since a concept can consist of an arbitrary amount of words, this can be overcome by formulating sufficiently specific concepts. Concepts are stored in a normalized format which aims at ignoring minor syntactic variations that do not affect the meaning of the concept. The following steps are used to normalize a concept [8]: remove punctuation, remove stop words, run each word through Porter's stemmer, alphabetize the stems, so that the order of words does not matter.

As mentioned earlier, ConceptNet focuses on concepts, rather than single words. Figure 2 (left) clearly shows that multi-word concepts (e. g., two, three, four words) form the largest part of the database, whereas single word concepts are significantly less common.

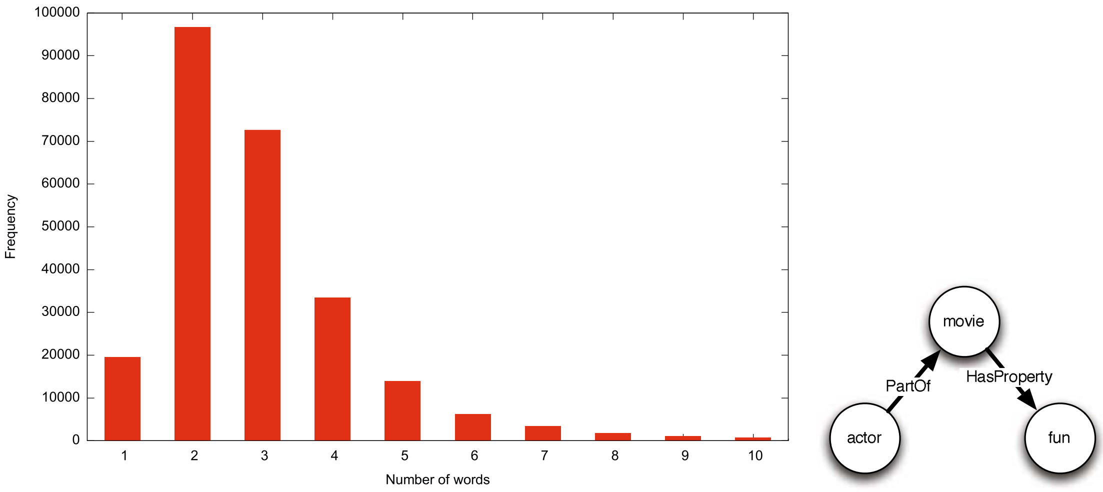

Fig. 2. The concept size in words and their occurrence frequency in ConceptNet 3 (left) and an example of concepts and relations found in ConceptNet 3 (right)

Concepts are interlinked by 21 different relations that encode the meaning of the connection between them. Relations are given intuitive names, e. g., `IsA` or `PartOf`. The unit of meaning representation is called a *predicate* in ConceptNet. An example of how predicates are stored in ConceptNet can be seen in figure 2 (right). Each predicate always consists of two concepts and one relation, e. g., *"actor"* `PartOf` *"movie"* (*"An actor is part of a movie"*). However, each concept can be part of many relations. In the example above, *"movie"* is also connected to *"fun"* by a `HasProperty` relation. To account for the fact that in the majority of cases predicates are not true if the order is changed (*"A movie is part of an actor"* does not make sense), relations are always unidirectional. They may also be negated to express that a certain statement is not true, such as *"A car cannot travel at the speed of light"*. Furthermore, each predicate has a score that represents its reliability. It defaults to 1 and can be increased/decreased by users. Predicates with zero or negative score are considered unreliable [8].

ConceptNet is available for download on the project homepage[7]. We use the snapshot from November 9, 2008 of the ConceptNet 3 database in the experiments reported in this chapter. This version comes with a total of 250 556 concepts, and 390 885 predicates for the English language.

General Inquirer [23] is a lexical database that uses tags to carry out its tasks. Each entry consists of the term and a number of tags denoting the presence of a specific property in the term. The two tags `Positiv` and `Negativ` express valence, hence they are the most interesting for our task. There are 1 915 terms in the `Positiv` category, and 2 291 in the `Negativ` counterpart. Examples are *"adore", "master", "intriguing"* for `Positiv`, and *"accident", "lack", "boring"* for `Negativ`. There is only partial support for part-of-speech information, to the extent that some terms have tags for different verb or adjective classes assigned. However, part-of-speech tags are not mandatory, and thus not all verbs or adjectives are tagged as such. General Inquirer also contains definitions and occurrence frequencies for most ambiguous words, which could be used to perform rudimentary word sense disambiguation.

WordNet is a database that organizes lexical concepts in sets of synonymous words, called *synsets*. Its design is inspired by current psycholinguistic and computational theories of human lexical memory [6]. Entries are strictly separated by syntactic category membership such as nouns, verbs, adjectives, and adverbs. Unlike ConceptNet, synsets are not linked by relations that express commonsense knowledge. They are rather connected by their lexical or semantic relatedness, such as *hyponymy* (one word being a specific instance of a more general word), *meronymy* (one word being a constituent part of another one), or *antonymy* (one word being the opposite of another). However, some of these relations are also found in ConceptNet, e. g., the complement of meronymy is `PartOf`.

4 Valence Estimation Using On-line Knowledge Sources

This section introduces the on-line knowledge sources-based approach to review valence estimation that has been developed during the work described in this chapter. Figure 3 shows the main building blocks of the algorithm. After preprocessing, two parallel steps extract words that convey sentiment information, as well as their targets – the words towards which the sentiment is expressed. This information is combined into expressions, which are in turn filtered in order to discard the irrelevant ones. A score value is obtained from the remaining expressions, which acts as the sole value upon which classification is performed. The remainder of this section gives a detailed description of each processing step.

Except for stemming (achieved by Porter's Snowball Stemmer [20]) and stopping, we used the implementations provided by OpenNLP[8]. The employed sentence detector utilizes a maximum entropy model to identify end of sentence characters. The Part-of-speech (POS) tagger used in our work is a stochastic

[7] http://conceptnet.media.mit.edu/

[8] http://opennlp.sourceforge.net/

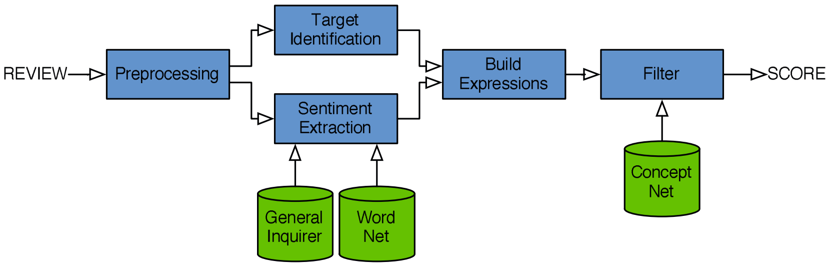

Fig. 3. Flowchart of the on-line knowledge sources-based approach

tagger based on maximum entropy. It supports the 48 word-level tags introduced by the Penn Treebank project [15]. Further, we use an English chunker: Each chunk is usually equal to a phrase in the syntax of a sentence, such as a noun phrase (NP), verb phrase (VP), or prepositional phrase (PP). The chunker is based on maximum entropy and the chunking model introduced in [22], which uses conditional random fields.

In order to transform the text into a suitable format for the main algorithm, some preprocessing steps are carried out. First, the text is split into sentences, which are in turn analyzed by a syntactic parser to label words and phrases with corresponding part-of-speech (POS) information. Since it is not necessary for our task to have comprehensive knowledge of the syntax, we employ a chunker only. An additional benefit of that is the flat structure produced by a chunker, which is better suited for the processing steps that follow.

As a unit of sentiment representation, we extract *ternary expressions (T-expressions)* on a per-sentence basis. T-expressions were introduced in [11] for automatic question answering tasks and have been adapted to product review classification in [29]. In the latter context, a T-expression has the following format: `<target, verb, source>`. Target refers to a feature term of the topic of the text, i. e., a movie in our case. The verb is selected from the same phrase in which the target occurs. If the verb does not convey sentiment information by itself, a sentiment source, which is typically an adverb, is extracted from the same phrase. Thus, the T-expression for the phrase *"a/DT well/RB written/VB story/NN"* is `<story, written, well>`. However, there are common situations where verbs are absent from a phrase, e. g., *"a/DT great/JJ movie/NN"*. Since a T-expression cannot be built in this case, we fall back to a second form, referred to as *binary expression (B-expression)* as suggested in [29]. A B-expression is simply a combination of an adjective and a target co-occurring in the text. Hence, the aforementioned utterance is represented by `<movie, great>`.

Target candidates are extracted from noun phrases (NP), based on the observation that feature terms are nouns. In [29], base noun phrase (BNP) patterns are applied successfully to extract sentiment about a given topic. A noun phrase needs to conform to one of these patterns in order to be considered for target identification. Patterns used are as follows: `NN`; `NN, NN`; `JJ, NN`; `NN, NN, NN`; `JJ, NN, NN`; `JJ, JJ, NN`. Before they are applied, words from POS classes that do not contribute to target identification are removed from the phrase. These are determiners, adverbs, and punctuation. All mentions of the personal pronoun *"it"* are regarded as referring directly to the topic of the text, and are also used as targets.

In the next step, the goal is to identify sentiment sources for each target, i. e., words that convey the actual affect information. In order to make sure that a given sentiment source is actually directed to the target in question, we need to restrict the search space. We accomplish this by finding 'border indicators' that appear between clauses or phrases within a sentence, and thus split it into units of statements. A sentiment source is only associated to a given target if they occur in the same section, i. e., there is no border indicator between them. Subordinating conjunctions, prepositional phrases, coordinating conjunctions, and punctuation such as commas or colons are used as border indicators.

We select all verbs and adjectives from the target section, and use General Inquirer to determine its valence v. A word u_i is assigned a valence $v(u_i) = 1$ if it has the General Inquirer tag `Positiv`, and a valence $v(u_i) = -1$ if it is tagged `Negativ`. In case a word is not found in General Inquirer, we use WordNet synsets to also lookup a word's synonyms, until a match is found. We refer to words for which a valence was found as 'sentiment words'. If a sentiment word is an adjective, a B-expression is built from it and stored in the result set. If it is a verb, its *siblings* (the words next to it within the same phrase) are first searched for sentiment adverbs. In case a match was found, we build a T-expression of the form `<target, verb, adverb>`. The adverb part is left out if no match was found. This step is also carried out for non-sentiment verbs, and a T-expression is generated if a sentiment adverb was found within its siblings. That way, phrases like *"a/DT beautifully/RB directed/VB movie/NN"*, with *"directed"* not being a sentiment verb, also yield an expression. Depending on the POS of the sentiment word, T-expressions and B-expressions are built.

We use the distance between the sentiment word and the target of an expression as a measure for its strength. It appears intuitive that words that occur closer together are more related. Many researchers have set a maximum distance between two words to decide whether or not they are related [16,25]. However, the experiments carried out in [30] showed that setting a maximum distance ('window size') actually degrades performance compared to not using it. Another approach that takes distance information into account was recently presented in [4]: Feature-opinion pairs similar to B-expressions are extracted, and the sentiment value of each pair is calculated using the multiplicative inverse of the distance between the two words. The results in that work also confirm that setting a maximum distance has a negative effect on performance.

Based on these findings, we use distance information for two purposes: First, for a given sentiment word, only the expression with the shortest distance between the sentiment word and its target is kept for further processing. The underlying assumption is that a sentiment word is in most cases directed at the target that it occurs closer to. That way we also lower the chance that a sentiment word is accidentally associated with an unrelated target. However, multiple expressions may exist for a given target, in order to catch all sentiment

words in sentences containing more than one. The sentence *"a/DT well/RB written/VB ,/, beautifully/RB directed/VB story/NN"* contains two sentiment adverbs: *"well"* and *"beautifully"*. Two corresponding T-expressions are extracted accordingly:
`<story, written, well>` and `<story, directed, beautifully>`.

Secondly, we also use the word distance to boost or lower the score of an expression by means of a decay function. The weighted score s of an expression that contains the target t_i and the sentiment source u_i is calculated according to the following equation:

$$s(u_i, t_i) = c \cdot v(u_i) \cdot \frac{1}{D(u_i, t_i)^e} \tag{1}$$

The valence of u_i, as taken from General Inquirer, is denoted by $v(u_i)$, and $D(u_i, t_i)$ is the distance of words between u_i and t_i. The decay function is similar to the one introduced in [4], as it is based on the multiplicative inverse of the distance, too. However, we added a constant factor c, and an exponent e to adjust and fine-tune the level of decay. Because

$$\frac{1}{D(u_i, t_i)^e} = 1 \tag{2}$$

holds for any e if $D(u_i, t_i) = 1$, i. e., u_i and t_i occur right next to each other, $c > 1$ effectively amplifies the valence in that case. On the other hand, c has little effect for $D(u_i, t_i) \gg 1$, which is the case for words that occur further apart and are therefore less related. By choosing $e > 1$, it is possible to let the valence decrease 'faster' for greater $D(u_i, t_i)$, thereby controlling the amount of influence of the word distance. In an analogous manner, $e < 1$ results in a 'slower' decrease of valence. We provide an evaluation for these parameters in our experiments in section 5.

4.1 Fallback Mechanisms

Due to the fact that it is not always possible to find an expression in a sentence, we rely on two fallback mechanisms that allow us to create pseudo expressions in case no real expression was found. The first mechanism is used if no target could be extracted from a sentence. In that case, we assume that the sentence is referring to the movie in question, and build pseudo B-expressions that have *"it"* as target. The search for sentiment words is extended to include verbs, adjectives, adverbs, and nouns. Since a real target is not available, we do not have word distance information for a pseudo expression. Therefore, the decay function cannot be applied, resulting in a score that is equal to the valence of the sentiment word that was extracted. Hence, this fallback mechanism is basically similar to keyword spotting. The second mechanism takes place if no target, and no sentiment words could be found in a sentence. Based on the observation that there are more than three times as many positive than negative reviews, the score of a sentence is set to $s = 1$ as a last resort. However, this case is rare, with only 1.9 % of the reviews in our test set not yielding any expressions. Failure to extract sentiment words can be caused by very short utterances, and the use of colloquial language: Colloquial terms are most likely not contained in general purpose dictionaries.

4.2 Filtering and Classification

The final step of our algorithm determines if the expressions that were found earlier are actually directed at the movie. We assume that an expression refers to the movie in question, if its target word is a feature of the term *"movie"*. Since we operate only in the movie domain, a manually assembled list of features could be used. However, building such a list can be a daunting task, and one of our intentions was to keep the algorithm domain-independent. Instead, we rely on ConceptNet to identify features. A drawback of this approach is that ConceptNet does not contain features specific to a given movie, as it has been designed to be a generic knowledge source. More precisely, it generally does not contain named entities such as names of actors, characters in the movie, or movie titles. Therefore, sentiment expressed against such named entities, e. g., "Al Pacino's performance was outstanding" are currently not extracted. We plan to overcome this issue by using an even larger scale encyclopedia that contains this kind of domain-specific knowledge for many different domains.

Assuming that the topic of a review (i. e., movie in our case) is known, feature terms can be selected by looking up the following predicates in ConceptNet: *feature* `PartOf` *"movie"*, *feature* `AtLocation` *"movie"*, and *"movie"* `HasProperty` *feature*. The features retrieved this way form the list against which expressions are checked. All expressions whose target is not contained in the feature list are filtered out. In case all expressions for a review would have to be dropped, filtering is not performed. This ensures that there is always at least one expression for each review.

In order to determine the final valence of a review, the accumulated score S of the n expressions it contains is calculated:

$$S = \sum_{i=1}^{n} s(u_i, t_i) \tag{3}$$

The valence class V is chosen as follows:

$$V = \begin{cases} +1 & \text{if } S \geq 0 \\ -1 & \text{if } S < 0 \end{cases} \tag{4}$$

The decision of including $S = 0$ in the positive class is simply motivated by the fact that there are more positive than negative reviews.

5 Evaluation

This section describes the experiments we conducted to assess the performance of the introduced approach in comparison to data-based learning by 'Bag of N-Grams' (BoNG) as extension over Bag of Words [9], i. e., constructing a vector space by one feature per N-Gram (rather than word) in the training by different normalizations of this N-Gram's frequency in the current review and assigning valence by Support Vector Machines (SVM) or Regression (SVR). Both methods have been optimized on binary classification into positive and negative reviews. Later on in this section, we study classification performance on all three classes, as well as regression performance on the full $0-100$ scale of scores that Metacritic provides. We will also investigate out of vocabulary resolution as well as ways of fusing information from both approaches to improve performance.

We rely on Metacritic's release year attribute to split the database (cf. table 3) into a training subset (odd years, 49 698 instances), and a test subset (even years, 52 924 instances). For parameter tuning, we simplify the task into a binary classification problem by dropping all instances of the *mixed* class from both the training (including development) and test data. The remaining data sets have 33 942 instances for training, and 36 094 instances for testing. In order to avoid overfitting, parameter tuning is performed using a development subset of the training data for validation. It is defined as every other odd year, starting at 1; or more precisely all years for which $(year - 1) \bmod 4 = 0$. There are 15 730 instances in the development set and the remaining 18 212 instances are used for training during development. Using this approach, we can eventually use the test set to evaluate the classifiers on unseen data.

As mentioned in section 2, there are a lot more positive than negative review instances in the database. This can be a problem when training a machine learning algorithm, because the resulting model would be biased towards the class that has more training instances. The recognition rate for the class with less instances would then be degraded. In order to avoid a biased model, we use down sampling without replacement to extract a subset of the training data set with a balanced class distribution. The resulting balanced training corpus has a size of 15 063 instances. The balanced subset that is used for training during development has 8 158 instances.

5.1 Parameter Tuning

Decay Function. We first carried out experiments to find the optimum parameters for the decay function as described. Various combinations of the two parameters c and e have been evaluated on the training data subset. The tested range is $c, e \in \{0.1, 0.2, ..., 1.0, 2.0\}$. First, we set $c = 1$ to determine the optimum value for e. Lowering e to values below 1 increased accuracy until a maximum was reached for $e = 0.1$ and $e = 0.2$. For all values of $e > 1$, accuracy decreased significantly [7]. Using $e = 0.1$, we determined the best setting for c. It turned out that setting c to values other than 1 had a negative effect on accuracy. Compared to the decay function used in [4], which is similar to $c = 1$ and $e = 1$, we were able to improve accuracy by 0.23 % for $c = 1$ and $e = 0.1$. Figure 4 shows a three-dimensional plot of the accuracy over the parameters c and e. The highest accuracy of 70.29 % for $c = 1.0$ and $e = 0.1$ is visible as a 'summit'. Significantly lower accuracy for $e > 1.0$ can be observed as a 'valley'. The 'ridge' indicates that $c = 1.0$ is the best setting for all values of e.

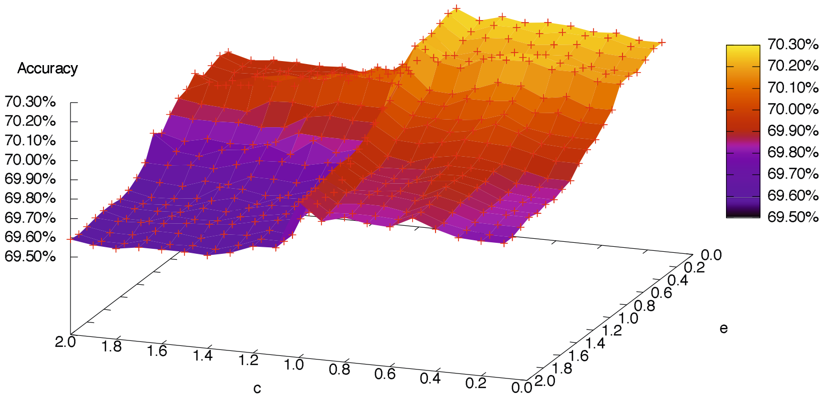

Fig. 4. Accuracy by different combinations of the decay function parameters c and e

Bag of N-Grams. For classification of the features we employ an implementation of the sequential minimal optimization algorithm described in [18] for training a support vector classifier using polynomial kernels [28].

With a size of over 83 k words, the vocabulary of the Metacritic database is large. After stemming is applied, over 62 k word stems remain. When using Bag of Words [9], this results in a feature space of the same size. If N-Grams are involved, the feature space can be significantly larger, depending on the minimum and maximum N-Gram lengths that are used. In order to make this problem computable, we need to take measures to reduce the dimensionality of the feature space. A minimum term frequency f_{min} can be employed to remove infrequent words. However, in [26] it is pointed out that low-frequency words are likely to be meaningful features for opinionated sentences. Since the valence of a review is determined by the opinions it contains, low-frequency words are indeed important for our task. Therefore, we use a gentle value of $f_{min} = 2$. With respect to this, we utilize a related method, *periodic pruning*, to carefully reduce the number of features without dropping potentially important ones. Periodic pruning divides the data set into chunks of configurable size, and is executed after such a chunk has been processed by the word or N-Gram tokenizer. Each feature that has only occurred once until this point is removed from the feature vector. If the chunk size is set high, the chances of discarding important features can be minimized. We chose to set it to 25 % of the data set.

Before the actual N-Gram parameters were evaluated, we determined the optimum settings for transformation and normalization. Table 4 gives the accuracy and F-measure obtained by each possible combination of methods. Simple N-Gram frequency (f_{ij}), term frequency transformation (TF), inverse document frequency transformation (IDF), and normalization (norm) are considered. The N-Gram parameters are set to $g_{min} = 1$ and $g_{max} = 3$, the setting that yields highest accuracy as we will show in the next section. Although the values do not differ much, normalization combined with TFIDF yields best results. Normalization increases performance in every case. IDF seems to benefit largely from normalization, as it only adds to performance significantly if values are normalized, too.

Table 4. Influence of transformations and normalization on Bag of N-Grams. Reported are overall accuracy and weighted F-measure.

Transformation	**Accuracy**	**F-measure**
f_{ij}	75.03 %	76.93 %
norm(f_{ij})	75.72 %	77.49 %
TF	75.44 %	77.30 %
norm(TF)	75.66 %	77.44 %
IDF	75.05 %	76.95 %
norm(IDF)	76.42 %	78.11 %
TFIDF	75.45 %	77.30 %
norm(TFIDF)	**77.14 %**	**78.57 %**

To investigate the influence of the N-Gram length, different combinations of g_{min} and g_{max} are tested. As shown in table 5, best accuracy is obtained for $g_{min} = 1$ and $g_{max} = 3$, and significantly decreases for $g_{min} > 1$. On the one hand, this is in line with the results reported in [3] for product reviews, where best performance was attained using trigrams. On the other hand, the same work reports that back-off N-Grams degrade performance, which we cannot confirm. Our results point out the importance of single words, which is emphasized further when looking at the composition of the feature space in case of the optimal settings. Only 12 % are taken by single words, whereas 52 % are bigrams, and 36 % are trigrams. This leads to the conclusion that despite their small contribution to the feature space, single words convey important valence information in the context of movie reviews in Metacritic. However, information about the context of a word, which is included by higher-order N-Grams, seems to be equally important. The largest feature space among our results has a dimensionality of 177 733 for $g_{min} = 1$ and $g_{max} = 5$. The smallest space for $g_{min} = 1$ and $g_{max} = 1$ has a size of 18 316 dimensions.

Table 5. Effect of the minimum and maximum N-Gram length g_{min} and g_{max} on the accuracy and weighted F-measure achieved by Bag of N-Grams

g_{min}	g_{max}	**# Features**	**Accuracy**	**F-measure**
1	1	18 316	74.58 %	76.43 %
1	2	96 152	75.95 %	77.70 %
1	**3**	**151 083**	**77.14 %**	**78.57 %**
1	4	171 438	76.41 %	78.03 %
1	5	177 733	76.92 %	78.46 %
2	2	77 840	66.72 %	69.53 %
2	3	132 780	66.54 %	69.38 %
2	4	153 146	69.62 %	72.07 %
2	5	159 465	71.66 %	73.73 %
3	3	54 968	71.59 %	73.43 %
3	4	75 418	72.33 %	74.05 %
3	5	81 911	72.61 %	74.29 %

In [17] it is reported that best movie review classification performance is obtained by using word presence information (as opposed to word frequency), and single words without stemming as features for SVM. Extensive experiments with different combinations of machine learning algorithms, N-Gram settings, stemming, and derived features such as part-of-speech and word position information are conducted in this work. Using the best reported combination, we obtain an accuracy of 76.51 %, and an F-measure of 77.97 % on our data. Comparing these values to our best results in table 5, the accuracy is 0.63 % lower, and the F-measure is 0.60 % lower when using the parameters suggested in [17].

5.2 Out of Vocabulary Resolution

If a given word is contained in the vocabulary of the test data set, but does not occur in the training vocabulary, we call it an *out of vocabulary* (OoV) word. Because these words are missing in the training set, a machine learning algorithm is unable to determine a numerical feature for them. Meaningful features are possibly lost due to this problem. Compared to the database vocabulary size of 83 328, the number of these out of vocabulary words is quite significant, adding up to 25 217, or 30.3 % of the vocabulary. The number of OoV events, i. e., the number of occurrences of such words in all reviews, is 38 244. This amounts to only 3.0 % of the total 1 288 384 words in the database (cf. table 6). The difference between the percentage of OoV vocabulary words and OoV events is likely caused by proper nouns, e. g., the title of a movie, or the name of its actors. Many of these words occur only in reviews for one single movie. Since the training and test data sets are obtained by separating movies released in odd years from those released in even years, all reviews for a given movie are contained in one set only. Therefore, these proper nouns are inevitably out of vocabulary.

Other forms of OoV can be reduced by utilizing a stemming algorithm. Because stemming treats all different forms of a word as one feature, OoVs resulting from a unique occurrence of a certain word form can be avoided. After stemming has been applied, the database vocabulary has a size of 62 212, and the OoV words account to 19 228. Surprisingly, the percentage of 30.9 % is actually slightly higher than before. Looking at OoV events however, the number decreases to 29 482, or 2.3 %, respecitvely, of all words.

Table 6. Number of OoV words and percentage of the vocabulary size, as well as number of OoV events and percentage of words in the database. Values are reported for no OoV resolution (baseline), stemming, and stemming with on-line knowledge sources-based (OKS) resolution.

# Words	Vocabulary	OoV words	OoV events
Baseline	83 328	25 217 (30.3 %)	38 244 (3.0 %)
Stemming	62 212	19 228 (30.9 %)	29 482 (2.3 %)
Stemming & OKS	62 212	14 123 (22.7 %)	22 746 (1.8 %)

In addition to stemming, OoVs can be resolved by replacing them with synonyms which are not OoV. We leverage the on-line knowledge sources ConceptNet and WordNet to find synonyms. If one of a word's synonyms does occur in the training vocabulary, it is used to replace every occurrence of the original word in the text. This step is carried out before N-Grams are assembled. We were able to substitute 5 105 vocabulary words and 6 736 OoV events using this method. Only 1.8 % of all words are still accounted as OoV, hence we were able to resolve 40.5 % of the initial OoV events using both stemming and synonym substitution.

Running the SVM classifier with both resolution methods enabled yields an accuracy gain of 0.04 %. Obviously, the resolved OoVs do not add a significant amount of important discriminating features. A possible cause is that the method of finding synonyms has a certain degree of inaccuracy. In fact, WordNet has no support for word sense disambiguation, and sometimes produces more noise than signal when used for finding similarities of meaning, according to [3]. Because a word can be part of multiple synsets, depending on its various meanings, they need to be equally considered. Therefore false correlations are very likely to be encountered. If in the worst case an OoV word is replaced by a false synonym which conveys a different sentiment, the overall accuracy suffers. Since this is a general shortcoming of WordNet, resolving it is beyond the scope of this chapter.

5.3 Bag of N-Grams vs. On-line Knowledge Sources

To compare the performance obtained by Bag of N-Grams to the approach based on on-line knowledge sources, we evaluated the algorithm described in section 4 on the same test data. The parameters for Bag of N-Grams are chosen as $g_{min} = 1$ and $g_{max} = 3$ to produce best results. Normalization and TFIDF transformation, as well as out of vocabulary resolution are used.

Results of the comparison between Bag of N-Grams and SVM (BoNG/SVM) and the on-line knowledge sources (OKS) based approach are shown in table 7. BoNG/SVM performs significantly better than OKS according to all reported measures. The accuracy of BoNG/SVM is 7.95 % higher than the accuracy of OKS. The largest gap between the two approaches is observed for the recall for

Table 7. Bag of N-Grams (BoNG) with SVM, Random Forests (RF), Naïve Bayes (NB), On-line Knowledge Sources (OKS), and Pointwise Mutual Information (PMI-IR) performance compared for two review classes (positive (+)/negative (-))

Measure	BoNG/SVM	BoNG/RF	BoNG/NB	OKS	PMI-IR
Accuracy	77.37 %	76.41 %	67.45 %	69.42 %	65.13 %
Weighted F-measure	78.77 %	77.94 %	69.12 %	70.29 %	65.91 %
Area under curve	0.777	0.852	0.643	0.644	0.605
$Precision_+$	92.18 %	92.12 %	82.78 %	81.92 %	78.54 %
$Precision_-$	50.84 %	49.52 %	35.69 %	36.70 %	28.11 %
$Recall_+$	77.07 %	75.76 %	72.73 %	77.21 %	75.09 %
$Recall_-$	78.39 %	78.58 %	50.01 %	43.67 %	32.19 %

negative reviews, which is 34.72 % lower for OKS. Obviously, negative reviews contain certain linguistic properties that are unaccounted for by this approach. A possible cause is the fact that our method does not detect if negation is involved in an expression it has extracted, assuming that negation is more common in negative reviews. Negation detection is a complex problem to solve [3,27], especially for syntax-driven approaches. BoNG solves this problem in an elegant way: By using N-Grams as features, a given term automatically results in a different feature if a negation word co-occurs within the neighbourhood range.

Interestingly, the precision for negative reviews is far below the precision for positive reviews in both approaches, meaning that they both tend to detect negative sentiment when it is actually positive. This confirms the hypothesis brought up in [24] that positive movie reviews may contain negative wording, e. g., to describe unpleasant scenes in a war or horror movie. To resolve this issue, domain specific knowledge of movie genres would be required.

Figure 5 (left) compares the receiver operating characteristic (ROC) of the two approaches. The values for BoNG were obtained by gradually shifting SVM's decision rule based on the distance to the class separating hyper plane. For OKS, the discrimination threshold d of the classifier was varied over the full range of scores observed for the test data. Scores range from 12.9 on the highest, to -7.6 on the lowest end. The step size between two consecutive thresholds was chosen to be 0.1. Some interesting values of d are highlighted in figure 5 (right). There are significant differences in the TPR and FPR between $d = 1.0$ and $d = 1.1$, as well as between $d = -0.9$ and $d = -1.0$. A score of ±1.0 corresponds to a review that has only a single positive or negative expression. Obviously a large number of such reviews exists in the database, which explains these significant

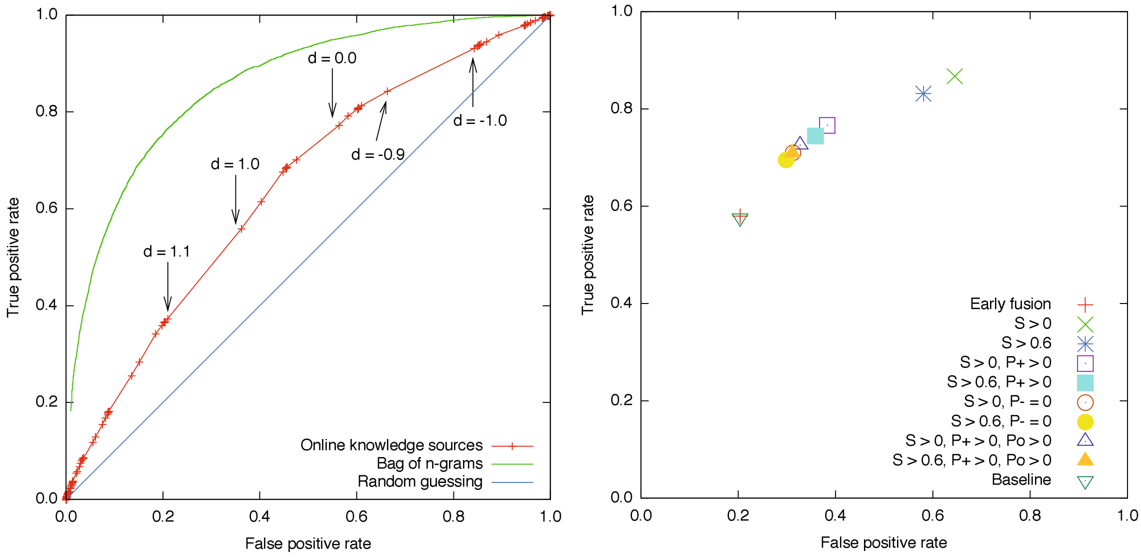

Fig. 5. ROC of OKS and BoNG/SVM for the two class problem (left). The discrimination threshold d is varied along the curve of OKS. The curve that results from random guessing is plotted for comparison. ROC for late fusion methods compared to the baseline without fusion (right). Values shown apply to the positive class.

differences. Also highlighted is $d = 0.0$, which is the discrimination threshold that was used in all experiments on the binary classification problem. The superior performance of BoNG is clearly visible in the graph, its ROC curve being more convex and closer to the upper left corner than the curve for OKS. In numerical terms, the area under curve for BoNG resembles 0.777, whereas the value for OKS is only 0.644. Also plotted is the ROC that results from randomly guessing on the class, which naturally has an area under curve of 0.5.

Since the two approaches are different in nature, we compared each of them to a more similar one. The key difference between them is that OKS is based on generally available knowledge sources, whereas BoNG is based on machine learning – usually employing data from the target domain. Therefore, we chose to compare OKS to the PMI-IR-based method described in [24], as it also incorporates data available on-line. The original algorithm uses the Altavista[9] search engine to get the number of matching web pages for a given query ("hits"), which is used as input data for the algorithm. We chose Yahoo![10] instead, as it provides a larger search index, more accurate hits numbers, and a fast API. The PMI-IR algorithm does not perform as good as OKS on our data set. It is able to achieve an accuracy of 65.13 %, which is 4.29 % short of OKS. PMI-IR and OKS both suffer from bad precision and recall for negative reviews. One of the reasons is that PMI-IR was not able to extract any phrases from 16 448 reviews in the test data set, which is 45.57 % of the instances. As a fallback, we assumed positive sentiment for these cases. The authors report an average of 29.13 phrases per movie review in their corpus. Since Metacritic reviews are very brief, it only extracted an average of 1.51 phrases per review on our data, not counting the reviews that did not yield any phrase.

Using the same N-Gram features, we compared the performance of the SVM classifier to other popular methods. We chose the simple yet widely used Naïve Bayes (e. g., [3]), as well as the more recent Random Forests [1] classifiers. We trained a forest with 100 trees, allowing infinite depth of the trees, as found to be optimal in a series of experiments on the development set. The number of features to consider for random feature selection was set to $F = int(log_2 M + 1)$, where M is the number of inputs [1]. BoNG/RF performs almost as good as BoNG/SVM, with an accuracy of 76.41 % only 0.96 % short of BoNG/SVM. BoNG/NB on the other hand falls back significantly, yielding only 67.45 % accuracy.

5.4 Error Analysis

Sentiment word detection yielded a total of 339 099 lookups, at a coverage of 65.72 %. With 4 206 words tagged either positive or negative, General Inquirer contains a relatively large amount of sentiment words. That said, one would expect a higher coverage rate. We suspect that the use of slang in the database, as well as the uncertainty involved in POS tagging are the main reasons for the low coverage rate. In the feature detection step, there were a total of 109 238

[9] http://www.altavista.com/

[10] http://www.yahoo.com/

feature lookups, at a coverage of 84.43 %. Obviously, the coverage of the on-line knowledge sources the algorithm is based on is problematic. We expect that performance of our algorithm will increase with future expansions of these data sources.

Another source of error are reviews that contain both positive and negative statements. Ideally, the algorithm would extract both statements, decide which one is stronger, and pick that one to determine the score. These cases are problematic, as the decision is not always obvious, even for a human being. In fact it is left to the author to decide on that. The following reviews taken from the corpus illustrate the problem.

"Completely ridiculous, but fun to look at." (Score: 25). This review contains a negative statement – "completely ridiculus"; and a positive one – "fun to look at". The author regards the negative one as being stronger, as he assigned a score of 25. The OKS algorithm correctly identifies "fun" and "ridiculous" as sentiment-bearing words, but also extracts "completely" as a positive word. Because it is not able to determine which statement is stronger, they are all equally considered, resulting in an overall score of 1.0 and leading to misclassification.

"Cheerful and easy to watch but surprisingly inept in the telling." (Score: 63). In this case, the algorithm is able to detect the sentiment correctly. It extracts two positive sentiment words – "easy" and "cheerful", and a negative one – "inept". This is in line with the expressed sentiment, and with the assigned score.

Another source of error are reviews that contain figurative or otherwise complex language. Because our approach is based only on shallow linguistic analysis, deeply buried meaning can often not be detected.

"Minimally plotted but beautifully atmospheric nightmare." (Score: 70). This review also contains a negative statement – "minimally plotted", as well as a positive one – "beautifully atmospheric nightmare". The words used in these two statements are problematic for the algorithm. From the first one, it extracts the verb "to plot", which has ambiguous meanings. Since there is no support for word sense disambiguation in General Inquirer, it is seen as a negative sentiment word. The second statement is problematic because of its use of figurative language. The true meaning cannot be detected by the algorithm – it yields a positive expression for "beautiful", and a negative expression for "nightmare". This results in an overall score of 0 for this statement. Because the first statement produced a negative score, the review is misclassified.

"Exploding on the screen in a riot of movement, music and color." (Score: 90). This review uses negative sentiment bearing words ("exploding", "riot") to express a positive opinion. The algorithm extracted only "exploding", putting this review in the negative class.

"The influence of Danish filmmaker Lars von Trier looms heavily over the whole film." (Score: 75). This review uses the verb "to loom" in order to express that the movie is dark, for which Lars von Trier is known. Lacking this background information, the algorithm only sees a negative sentiment word, which leads to misclassification.

"The less said the better." (Score: 25). Here, the algorithm only extracts the word "better" as a positive sentiment word. Again, shallow linguistic analysis cannot extract the true meaning of the sentence.

5.5 Estimating Mixed Sentiment

Until now, our evaluation has been carried out on a prototypical two class problem. However, the Metacritic database contains a third class for mixed or average reviews (see table 2). We skipped this class for parameter optimization in order to simplify the problem to a binary classification task, and because mixed or neutral reviews are especially hard to estimate [25,5], just as non-prototypical sentiment in general [21]. Both our approaches do not perform subjectivity/objectivity classification, which is widely used to distinguish neutral reviews from opinionated (positive/negative) ones. But still we were able to gain results by making only minor modifications to the algorithms. For the data-driven approach, the three class problem is modeled as a series of binary classification problems which are solved in the same way as before by SVM. For the syntax-driven approach, the decision function in (4) has to be replaced by one that supports three classes. The solution is to break down the three class problem into two binary classification problems. For the first classifier, the negative and mixed classes are treated as one class, and distinguished from the positive class. The second classifier distinguishes between negative and mixed/positive merged into one class. In order to obtain the best possible overall performance, we evaluate each classifier separately and observe two classification thresholds τ_+ and τ_-. The thresholds can then be combined into a valence decision function

$$V = \begin{cases} +1 & \text{if} \quad S > \tau_+ \\ 0 & \text{if} \quad \tau_- \leq S \leq \tau_+ \\ -1 & \text{if} \quad S < \tau_- \end{cases} \tag{5}$$

that distinguishes between all three classes. Because of $\tau_- < \tau_+$, we effectively allow a range of values around $S = 0$ to be associated to the mixed class. We observed the average weighted F-measure for optimizing the two binary classifiers on the validation data set. The highest values were obtained by setting $\tau_+ = 0.6$ for the first classifier, and $\tau_- = -1.9$ for the second. The performance of the resulting ternary classifier is shown in table 8, next to the performance of the BoNG ternary classifier. Both classifiers have been evaluated on the test data set.

Both approaches show significantly lower performance compared to the two class problem. BoNG accuracy drops by 23.66 % to 53.71 %, and OKS accuracy is decreased by 20.04 % to 49.38 %. Better values are achieved by the BoNG approach for most measures. Only the recall for positive reviews is higher when OKS is used, with 68.43 % compared to 57.62 % for BoNG. OKS, however, suffers from a low recall for negative instances. BoNG recall values are more balanced between the three classes, ranging from 43.91 % for mixed, up to 60.43 % for negative reviews.

Table 8. Bag of N-Grams (BoNG) and On-line Knowledge Sources (OKS) performance compared for three review classes (positive (+)/mixed (0)/negative (-)). The OKS classification thresholds are $\tau_+ = 0.6$, and $\tau_- = -1.9$.

Measure	**BoNG**	**OKS**
Accuracy	**53.71 %**	49.38 %
Weighted F-measure	**55.04 %**	47.19 %
$Precision_+$	**75.66 %**	58.82 %
$Precision_0$	**42.92 %**	35.74 %
$Precision_-$	**34.70 %**	28.71 %
$Recall_+$	57.62 %	**68.43 %**
$Recall_0$	**43.91 %**	37.32 %
$Recall_-$	**60.43 %**	10.66 %

5.6 Information Fusion

The evaluation of the two approaches on three review classes has shown that none of them performs always better than the other. Obviously, each method has advantages in certain areas, but disadvantages in other areas. By combining them into one classifier, the disadvantages may be alleviated. All experiments in this section again use the parameters which yielded best results earlier. The N-Gram settings are chosen as $g_{min} = 1$ and $g_{max} = 3$; normalization and TFIDF transformation, as well as out of vocabulary resolution are used.

Early fusion takes place at the feature level, before the machine learning algorithm is run, thus preserving utmost knowledge prior to the final decision process. This is known to be in particular advantageous if the information is highly correlated, as given here [14]. In our case, we can simply add the score produced by OKS to the BoNG feature vector, and then run SVM on the extended vector. The influence of the OKS score on the class decision is relatively low in this case, since it 'competes' with the numerous other components of the feature vector, which are generated by BoNG. The performance achieved by early fusion is given in table 9. Compared to the baseline values, i. e., the BoNG results given in table 8, accuracy increased slightly by 0.13 % to 53.84 %. There is an improvement in the recall for positive and negative reviews, however, only at the cost of a worsened recall for the mixed class.

In contrast to early fusion, late such combines the output of different methods on a semantic layer [14]. Each approach determines a numerical or boolean result, and the results are, e. g., arithmetically or logically combined to make the final decision on a class. This allows for selectively switching between the involved methods based on their performance characteristics, in order to obtain best overall performance. From the results presented earlier we know that the BoNG approach has better performance than OKS in almost all measures. However, OKS has a much better recall of positive reviews, if all three review classes are considered. The idea arises to improve overall recognition of positive reviews by relying on OKS, while basing the decision for other classes on BoNG only.

SVM are able to provide a prediction value for each class based on the distance to the separating hyperplane, on which the final classification is based. The prediction values for each class will be denoted as P_- (negative), P_0 (mixed), and P_+ (positive). Each value determines how much evidence the classifier has for the corresponding class. The range for these values is $0 \leq P \leq 1$, where 0 stands for weakest, and 1 stands for strongest evidence. Together with the score S produced by OKS, we can set conditions under which the classifier votes for the positive class. If the conditions are not met, SVM's initial vote is used to determine the class. The different conditions we evaluated, and the resulting performance, are shown in table 9. Regarding the OKS score, we examined both $S > 0$, and $S > 0.6$, the latter being the positive discrimination threshold τ_+ we determined earlier. For SVM we investigated evidence for the positive class being present ($P_+ > 0$), no evidence for the negative class ($P_- = 0$), and evidence for both the mixed and positive classes ($P_+ > 0,\ P_0 > 0$).

Table 9. Early and late fusion methods for the three review classes (positive (+)/mixed (0)/negative (-)) compared to the baseline (no fusion) with the conditions for the OKS score S, and the SVM predictions P_-, P_0, P_+ under which the classifier votes for the positive class are given

	Accuracy	*Recall*$_-$	*Recall*$_0$	*Recall*$_+$
Baseline	53.71 %	60.43 %	**43.91 %**	57.62 %
Early fusion	53.84 %	**60.67 %**	43.65 %	57.96 %
Late fusion				
$S > 0$	55.93 %	32.65 %	16.83 %	**86.72 %**
$S > 0.6$	55.95 %	37.04 %	20.60 %	83.14 %
$S > 0,\ P_+ > 0$	**57.77 %**	51.64 %	29.74 %	76.64 %
$S > 0.6,\ P_+ > 0$	57.37 %	52.74 %	31.56 %	74.45 %
$S > 0,\ P_- = 0$	56.19 %	60.43 %	29.74 %	70.98 %
$S > 0.6,\ P_- = 0$	55.99 %	60.43 %	31.56 %	69.49 %
$S > 0,\ P_+ > 0,\ P_0 > 0$	56.83 %	59.14 %	29.74 %	72.58 %
$S > 0.6,\ P_+ > 0,\ P_0 > 0$	56.54 %	59.25 %	31.56 %	70.89 %

All late fusion methods are significantly better in a one-tailed test at the 0.1 % level and can therefore be considered a true improvement. Early fusion improvement over the baseline is not significant and therefore can be considered as not to improve performance.

At first glance it is not obvious which combination of late fusion parameters has the best overall performance. On the one hand, best accuracy is obtained by $S > 0$ and $P_+ > 0$. On the other hand this configuration shows a low recall rate for positive and negative reviews at the same time. The most balanced recall over all three classes is obtained by the baseline and early fusion methods, however, they also produce a low accuracy of 53.71 % and 53.84 %, respectively. In order to have better measures for comparison, we rely on an ROC graph. The true positive rates (TPR) over the false positive rates (FPR) for all combinations from table 9 are plotted in figure 5 (right). The baseline and early fusion methods produce points towards the left part of the graph, and can therefore be regarded as 'conservative' classifiers. They have the lowest FPR, at the cost of the lowest TPR. Using only the OKS score produces 'liberal' classifiers, with points close to the top right corner of the graph. The TPR is the highest among the evaluated methods, but the FPR significantly increases as well. All combinations which are using values from both approaches, produce points between these extreme cases. Judging from the ROC graph only, best performance is obtained for the conditions $S > 0.6,\ P_+ > 0,\ P_0 > 0$ since its point in the graph appears closest to the top left corner.

5.7 Review Score Prediction

In order to distinguish even between reviews within a class, we next use Support Vector Regression to estimate the score value of reviews on its scale of $0 - 100$. Since SVR uses the same vector space representation as SVM, the best settings we determined earlier for Bag of N-Grams are used. Normalization and TFIDF transformation, as well as out of vocabulary resolution are also employed again. N-Gram parameters are chosen to be the optimum values $g_{min} = 1$ and $g_{max} = 3$. A radial basis function kernel with the variance parameter set to $\gamma = 0.01$ is further chosen based on findings from the development set. In order to evaluate the performance, two values are observed: The *correlation coefficient*, and the *mean absolute error*. We achieve a correlation coefficient of 0.57, and a mean absolute error of 14.1 on the test data. Occurring deviations from the actual scores may be owed to the fact that the score value is not evenly distributed within the data set (see figure 1).

When interpreting these results, one should keep in mind that the ground truth itself is likely to be partly inaccurate: The reviews in Metacritic are written by both professional and non-professional writers, and usually only the author of a review gets to decide on its score. It has been reported that multiple persons often disagree on which sentiment is expressed in a given review, or through a given word [17,3]. Methods that rely heavily on on-line knowledge sources are additionally impacted, because they suffer from the same inaccuracy of judgement. ConceptNet tries to overcome the problem by letting users vote on the reliability of predicates.

6 Conclusion and Outlook

Different evolution steps of on-line knowledge sources have been used, from annotated dictionaries (General Inquirer, 1966) to comprehensive semantic networks (ConceptNet, 2003). Two approaches suitable for valence estimation of movie reviews have been presented. However, the proposed methods are generic in nature and do not rely on domain-specific features. The on-line knowledge sources-based approach leverages linguistic methods, dictionaries, and semantic networks to estimate valence, and tries to resemble the way that humans understand text. Its special benefit is that it does not require any training. The data-driven approach is based on Bag of N-Grams and Support Vector Machines. On-line knowledge sources have been utilized to improve performance of this approach by resolving out of vocabulary events. While we were able to resolve 40.5 % of these events, the accuracy gain was only 0.04 %. Both methods have been optimized on a subset of Metacritic containing only positive and negative reviews. We were able to achieve 77.37 % accuracy with the data-driven approach, and 69.42 % with the on-line knowledge sources approach. Early fusion was applied, but could not improve the accuracy. Both our approaches show significant performance gains compared to previous state of the art methods. We later extended the problem to three valence classes by estimating mixed reviews as well. The data-driven approach was able to achieve 53.71 %, whereas the on-line knowledge sources approach achieved 49.38 %. The latter showed very unbalanced recall for the three classes, with 68.43 % recall for positive reviews. We were able to improve accuracy up to 57.77 % using late fusion. The method that showed the best compromise between the true positive and false positive rates has an accuracy of 56.54 %. The data-driven approach has been extended by Support Vector Regression. To evaluate this approach, we observed the correlation coefficient between the actual and the predicted score, which was 0.57. The mean absolute error was 14.1 on Metacritic's $0 - 100$ scale of scores.

Based on these observations, we propose the following future improvements: We are seeking to improve performance of the on-line knowledge sources-based approach by incorporating more specific term categories from General Inquirer. [13] have shown that ConceptNet can serve as a source of sentiment information as well. It could complement General Inquirer for that purpose. Furthermore, multi-word terms or even full statements could be extracted and looked up in ConceptNet. Together with named entity recognition, other on-line knowledge sources as Wikipedia will allow us to detect for example names of actors or characters in a movie, and use them as additional features. In order to further improve out of vocabulary resolution, more WordNet relations, e. g., *antonymy* (opposite meaning), or *hypernymy* (more general meaning) could be considered. Our method currently does not resolve out of vocabulary N-Grams. By adding a second substitution step after N-Grams have been assembled, this can be done as well. Finally, Metacritic's further review bodies dealing with music and video games can be considered for according experiments.

References

1. Breiman, L.: Random forests. Machine Learning 45(1), 5–32 (2001)
2. Das, S.R., Chen, M.Y.: Yahoo! for amazon: Sentiment parsing from small talk on the web. In: Proceedings of the 8th Asia Pacific Finance Association Annual Conference (2001)
3. Dave, K., Lawrence, S., Pennock, D.M.: Mining the peanut gallery: opinion extraction and semantic classification of product reviews. In: Proceedings of the 12th international conference on World Wide Web, pp. 519–528. ACM, Budapest (2003)
4. Ding, X., Liu, B., Yu, P.S.: A holistic lexicon-based approach to opinion mining. In: WSDM 2008: Proceedings of the International Conference on Web Search and Web Data Mining, pp. 231–240. ACM, New York (2008)

5. Esuli, A., Sebastiani, F.: Determining term subjectivity and term orientation for opinion mining. In: Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2006), Trento, Italy (2006)
6. Fellbaum, C.: Wordnet: An Electronic Lexical Database. MIT Press, Cambridge (1998)
7. Gillick, L., Cox, S.J.: Some statistical issues in the comparison of speech recognition algorithms. In: Proceedings of the International Conference on Audio Speech and Signal Processing (ICASSP), vol. I, pp. 23–26. Glasgow, Scotland (1989)
8. Havasi, C., Speer, R., Alonso, J.: Conceptnet 3: a flexible, multilingual semantic network for common sense knowledge. In: Recent Advances in Natural Language Processing. Borovets, Bulgaria (September 2007)
9. Joachims, T.: Text categorization with support vector machines: learning with many relevant features. In: Nédellec, C., Rouveirol, C. (eds.) ECML 1998. LNCS, vol. 1398, pp. 137–142. Springer, Heidelberg (1998)
10. Jurafsky, D., Martin, J.H.: Speech and Language Processing. Prentice-Hall, Englewood Cliffs (2000)
11. Katz, B.: From sentence processing to information access on the world wide web. In: Proceedings of the AAAI Spring Symposium on Natural Language Processing for the World Wide Web, pp. 77–86 (1997)
12. Liu, B., Hu, M., Cheng, J.: Opinion observer: analyzing and comparing opinions on the web. In: WWW 2005: Proceedings of the 14th International Conference on World Wide Web, pp. 342–351. ACM, New York (2005)
13. Liu, H., Lieberman, H., Selker, T.: A model of textual affect sensing using real-world knowledge. In: IUI 2003: Proceedings of the 8th International Conference on Intelligent User Interfaces, pp. 125–132. ACM, New York (2003)
14. Lizhong, W., Oviatt, S., Cohen, P.R.: Multimodal integration – a statistical view. IEEE Transactions on Multimedia 1, 334–341 (1999)
15. Marcus, M., Marcinkiewicz, M., Santorini, B.: Building a large annotated corpus of english: the Penn Treebank. Computational Linguistics 19(2), 313–330 (1993)
16. Morinaga, S., Yamanishi, K., Tateishi, K., Fukushima, T.: Mining product reputations on the web. In: KDD 2002: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 341–349. ACM, New York (2002)
17. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up?: sentiment classification using machine learning techniques. In: Proceedings of EMNLP 2002, Morristown, NJ, USA. Association for Computational Linguistics, pp. 79–86 (2002)
18. Platt, J.C.: Fast training of support vector machines using sequential minimal optimization, pp. 185–208. MIT Press, Cambridge (1999)
19. Popescu, A., Etzioni, O.: Extracting product features and opinions from reviews. In: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Morristown, NJ, USA, pp. 339–346 (2005)
20. Porter, M.F.: An algorithm for suffix stripping. Program 14(3), 130–137 (1980)
21. Schuller, B., Steidl, S., Batliner, A.: The interspeech 2009 emotion challenge. In: Proceedings of the Interspeech, Brighton, UK, pp. 312–315 (2009)
22. Sha, F., Pereira, F.: Shallow parsing with conditional random fields. In: NAACL 2003: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology. Association for Computational Linguistics, Morristown, NJ, USA, pp. 134–141 (2003)

23. Stone, P., Kirsh, J., Associates, C.C.: The General Inquirer: A Computer Approach to Content Analysis. MIT Press, Cambridge (1966)
24. Turney, P.D.: Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL), Philadelphia, pp. 417–424 (July 2002)
25. Turney, P.D., Littman, M.L.: Measuring praise and criticism: Inference of semantic orientation from association. ACM Transactions on Information Systems 21(4), 315–346 (2003)
26. Wiebe, J., Wilson, T., Bell, M.: Identifying collocations for recognizing opinions. In: Proceedings of the ACL 2001 Workshop on Collocation: Computational Extraction, Analysis, and Exploitation, pp. 24–31 (2001)
27. Wilson, T., Wiebe, J., Hoffmann, P.: Recognizing contextual polarity in phrase-level sentiment analysis. In: HLT 2005: Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Morristown, NJ, USA, pp. 347–354 (2005)
28. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques, 2nd edn. Morgan Kaufmann, San Francisco (2005)
29. Yi, J., Nasukawa, T., Bunescu, R., Niblack, W.: Sentiment analyzer: extracting sentiments about a given topic using natural language processing techniques. In: Proceedings of the Third IEEE International Conference on Data Mining, pp. 427–434 (November 2003)
30. Zhang, M., Ye, X.: A generation model to unify topic relevance and lexicon-based sentiment for opinion retrieval. In: SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval, pp. 411–418. ACM, New York (2008)
31. Zhuang, L., Jing, F., Zhu, X.Y.: Movie review mining and summarization. In: Proceedings of the 15th ACM International Conference on Information and Knowledge Management (CIKM 2006), pp. 43–50. ACM, New York (2006)