Chapter 6

Coarse-Grained Sentiment Analysis

Having familiarized ourselves with the peculiarities of the creation of a sentiment corpus, the different ways to automatically induce new polarity lists, and the difficulties of fine-grained opinion mining, we now move on to the presumably most popular sentiment analysis task—coarse-grained sentiment analysis or CGSA, in which we need to determine the overall polarity of a message.

Traditionally, this objective has been addressed with either of the three popular method groups:

- lexicon-based approaches,
- machine-learning–based (ML) techniques,
- and deep-learning–based (DL) applications.

In this chapter, we are going to scrutinize the most prominent representatives of each of these paradigms and also analyze their errors, the utility of single components, and the effect of additional training factors on their net results.

We begin our comparison by first presenting the metrics that we will use in our subsequent evaluation. After briefly describing the data preparation step, we proceed to the actual estimation of popular lexicon-, ML-, and DL-based approaches, explaining and evaluating them in Sections 6.3, 6.4, and 6.5 respectively. Finally, we conclude with an extensive evaluation of different hyperparameters and settings (including the impact of additional noisily labeled training data, various types of sentiment lexicons, and text normalization), summarizing our results and recapping our findings at the end of this part.

6.1 Evaluation Metrics

To estimate the quality of the compared systems, we will rely on two established evaluation metrics which are commonly used for measuring CGSA results: One of these metrics is the *macro-averaged F_1-score* over the two major polarity classes (positive and negative):

$$F_1 = \frac{F_{pos} + F_{neg}}{2}.$$

This measure has been first introduced by the organizers of the SemEval competition (Nakov et al., 2013; Rosenthal et al., 2014, 2015) and became a de facto standard not only for the SemEval dataset but virtually for all related coarse-grained sentiment tasks and corpora.

The second metric is the *micro-averaged F_1-score* over all three semantic orientations (positive, negative, and neutral), which essentially corresponds to the accuracy over the complete dataset (see Manning and Schütze, 1999, p. 577). This measure both predates and supersedes the SemEval evaluation as it had already been used in the very first works on coarse-grained opinion mining (Wiebe et al., 1999; Das and Chen, 2001; Read, 2005; Kennedy and Inkpen, 2006; Go et al., 2009) and has been again reintroduced recently at the GermEval shared task in 2017 (Wojatzki et al., 2017).

Furtermore, in addition to the above two metrics, we will also give a detailed information on the precision, recall, and F_1-scores of each particular polarity class in order to get a better intuition about precise strengths, weaknesses, and biases of each approach.

6.2 Data Preparation

Similarly to the data preparation steps used for the fine-grained sentiment analysis, we preprocessed all tweets involved in our experiments with the text normalization system of Sidarenka et al. (2013), tokenized them using the same adjusted version of Potts' tokenizer,[1] and lemmatized and assigned part-of-speech tags to these tokens with the `TreeTagger` of Schmid (1995). Moreover, like in the previous chapter, we automatically obtained morphological features for each word, and induced syntactic trees for each sentence with the `Mate` dependency parser (Bohnet et al., 2013).

We again divided the PotTS corpus (Sidarenka, 2016) into a training, development, and test set, using 70% of the tweets for learning, 10% for tuning and picking the optimal model parameters, and the remaining 20% for evaluating the results. We inferred the polarity labels for these microblogs with a simple heuristic rule akin to the one used by Wiebe and Riloff

[1] `http://sentiment.christopherpotts.net/code-data/happyfuntokenizing.py`

(2005), and assigned the positive (negative) class to the messages which had exclusively positive (negative) sentiments, skipping all microblogs that simultaneously contained multiple opinions with different semantic orientations. In cases when a sentiment was absent, we recoursed to a fallback strategy by considering all tweets with only positive (negative) polar terms as positive (negative), disregarding messages which featured expressions from both polarity classes, and assuming the rest of the corpus (i.e., posts with neither sentiments nor polar terms) to be neutral.

A few examples of such heuristically inferred labels are provided below:

Example 6.2.1 (Coarse-Grained Sentiment Annotations)
Tweet: Ich finde den Papst putzig ☺
I find the Pope cute ☺.
Label: **positive**

Tweet: typisch Bayern kaum ist der neue Papst da und schon haben sie ihn in der Tasche ...
Typical Bavaria The new Pope is hardly there, as they already have him in their pocket
Label: **negative**

As we can see from the examples, our simple rule provides fairly reasonable decisions. However, since this approach is still an approximation and consequently prone to errors (especially in the cases where the polarity of the whole microblog differs from the semantic orientation of its single tokens or is expressed without any explicit polar expressions at all, see Example 6.2.2), we also decided to evaluate all methods presented in this chapter on another German Twitter corpus which has been explicitly annotated with message-level polarities—the SB10k (Cieliebak et al., 2017).

Example 6.2.2 (Erroneous Sentiment Annotations)
Tweet: Unser Park, unser Geld, unsere Stadt! -NICHT unser Finanzminister! ☺ #schmid #spd #s21 #btw13
Our park, our money, our city! -NOT our Finance Minister! ☺ #schmid #spd #s21 #btw13
Label: **positive***

Tweet: Auf die Lobby-FDP von heute kann Deutschland verzichten ...
Germany can go without today's lobby FDP
Label: **neutral***

The SB10k dataset has been introduced by Cieliebak et al. (2017), and comprises a total of 9,738 tweets. These messages were sampled from a larger snapshot of 5M German microblogs

gathered between August and November 2013. To ensure lexical diversity and proportional polarity distribution in this corpus, the authors first split all posts of this snapshot into 2,500 clusters using the k-means algorithm with unigram features. Afterwards, from each of these groups, they selected tweets with at least one positive and one negative term from the German Polarity Clues lexicon (Waltinger, 2010). Each of these messages was subsequently annotated by at least three human experts from a pool of 34 different coders. The resulting inter-rater reliability (IRR) of these data run up to 0.39 Krippendorff's α (Krippendorff, 2007). Unfortunately, due to the restrictions of Twitter's terms of use (which allow to distribute only the ids of the microblogs along with their labels), we could only retrieve 7,476 tweets of this collection, which, however, still represents a substantial part of the original data.

In addition to the aforementioned two corpora (PotTS and SB10k), we also automatically annotated all microblogs of the German Twitter Snapshot (Scheffler, 2014), following the procedure proposed by Read (2005) and Go et al. (2009), and assigned the positive (negative) class to the tweets which contained the respective emoticons, regarding the rest of the microblogs as neutral.

The resulting statistics on the number of messages and polarity class distribution in these data are shown in Table 6.1.

Dataset	**Polarity Class**				**Agreement**	
	Positive	**Negative**	**Neutral**	**Mixed***	α	κ
PotTS	3,380	1,541	2,558	513	0.66	0.4
SB10k	1,717	1,130	4,629	0	0.39	NA
GTS	3,326,829	350,775	19,453,669	73,776	NA	NA

Table 6.1: Polarity class distribution in PotTS, SB10k, and the German Twitter Snapshot (GTS)
(* – the *mixed* polarity was excluded from our experiments)

As we can see, each dataset has its own unique composition of polar tweets: The PotTS corpus, for example, shows a conspicuous bias towards the positive class with 42 percent of the microblogs belonging to this polarity. We can partially explain this phenomenon by the following two reasons: first of all, this effect might be due to the coarseness of the heuristic rule that we applied to infer the labels for the messages; and, secondly, it might also stem from the initial selection criteria that we used to compile the data for this collection. As you might remember, a major part of this dataset was composed from the tweets which contained smileys or had at least one polar expression from the SentiWS lexicon (Remus et al., 2010). Since most of these emoticons were positive (which is evident from the statistics of the German Twitter snapshot), the selected posts also became skewed towards this semantic orientation.

The second most frequent group of the PotTS corpus are neutral microblogs, which account for 32 percent of the data. Negative messages, vice versa, represent a clear minority in this collection (only 19%), which, however, is less surprising as the same tendency can be observed for the SB10k and German Twitter Snapshot too.

Regarding the last two corpora, we can observe a more uniform (though not identical) behavior where both datasets are dominated by neutral posts, which constitute 62% of SB10k data and 84% of the German Twitter Snapshot. The positive class, again, makes up a big part of these datasets (23% in the former and 14% in the latter case), but its influence this time is much less pronounced than in the PotTS case. Finally, negative tweets are again the least represented semantic orientation. The only group which has even less instances than this class is the MIXED polarity. We, however, will skip the mixed orientation in our experiments for the sake of simplicity and uniformity of evaluation.[2]

Last but not least, the results of the inter-rater reliability test confirm the superior quality of the PotTS corpus, which, even despite its approximate labels, still has an α agreement (Krippendorff, 2007) that is almost 1.7 times as high as the respective score of the SB10k set (0.66 versus 0.39).

6.3 Lexicon-Based Methods

The first group of approaches that we are going to explore in this chapter using the aforementioned data are lexicon-based (LB) systems. Just like sentiment lexicons themselves, LB methods for coarse-grained opinion mining have attracted a lot of attention from the very inception of the sentiment analysis field. Starting from the work of Hatzivassiloglou and Wiebe (2000), who gave a statistical proof that the mere occurrence of a subjective adjective from an automatically compiled polarity list was a sufficiently reliable indicator that the whole sentence was subjective, more and more researchers started using lexicons in order to estimate the overall polarity of a text.

One of the first notable steps in this direction was made by Das and Chen (2001), who proposed an ensemble of five classifiers (two of which were purely lexicon-based and the other three heavily relied on lexicon features) to predict the polarity of stock messages (*buy*, *sell*, or *neutral*), achieving an accuracy of 62% on a corpus of several hundreds stock board messages. A much simpler method for a related task was suggested by Turney (2002), who determined the *semantic orientation* (SO) of reviews by averaging the PMI scores of their terms, getting these scores from an automatically generated sentiment lexicon. With this

[2] As we will see later, some of the CGSA methods (especially the lexicon-based ones) can hardly be extended to the prediction of more than three polarity classes.

approach, the author could reach an accuracy of 74% on a corpus of 410 manually labeled Epinions comments. In the same vein, Hu and Liu (2004) computed the overall polarity of a sentence by comparing the numbers of positive and negative terms appearing in it, reversing the orientation of the terms if they were negated.

In 2006, Polanyi and Zaenen presented an extensive overview and analysis of common lexicon-based sentiment methods that existed at that time, arguing that, besides considering the lexical valence (i.e., semantic orientation) of polar expressions, it was also important to incorporate syntactic, discourse-level, and extra-linguistic factors such as negations, intensifiers, modal operators (e.g., *could* or *might*), presuppositional items (e.g., *barely* or *failure*), irony, reported speech, discourse connectors, genre, attitude assessment, reported speech, multi-entity evaluation, etc. This theoretical hypothesis was also proven empirically by Kennedy and Inkpen (2006) who investigated two ways to determine the polarity of a customer review: In the first approach, the authors simply compared the numbers of positive and negative terms in the text, assigning the review to the class with the greater number of items. In the second attempt, they enhanced the original system with an additional information about contextual valence shifters, increasing or decreasing the sentiment score of a term if it was preceded by an intensifier or downtoner, and changing the polarity sign of this score to the opposite in case of a negation. With this adjustment, Kennedy and Inkpen achieved a statistically significant improvement, boosting the accuracy of the two-class prediction from 67.9 to 69.3%.

Finally, a veritably seminal work on lexicon-based techniques was presented by Taboada et al. (2011) who introduced a manually compiled polarity list[3] and used this resource to estimate the overall semantic orientation of the text. Drawing on the ideas of Polanyi and Zaenen (2006), the authors incorporated a set of additional heuristic rules into their computation by changing the prior SO values of negated, intensified, and downtoned terms, ignoring irrealis and interrogative sentences, and adjusting the weights of specific document sections. An extensive evaluation of this approach showed that the manual lexicon performed much better than automatically generated polarity lists including the Subjectivity Dictionary (Wilson et al., 2005), Maryland Polarity Set (Mohammad et al., 2009), and SentiWordNet of Esuli and Sebastiani (2006b). Moreover, the authors also demonstrated that their method could be successfully applied to other topics and genres, hypothesizing that lexicon-based approaches were in general more amenable to domain shifts than traditional supervised machine-learning techniques.

It is therefore not surprising that lexicon-based systems have also quickly found their way into the sentiment analysis of social media: For example, one such approach, explicitly tai-

[3]The authors hand-annotated all occurrences of adjectives, nouns, and verbs found in a corpus of 400 Epinions reviews with ordinal categories ranging from -5 to 5 which reflected the semantic orientation of a term (positive vs. negative) and its polar strength (weak vs. strong).

lored to Twitter specifics, was proposed by Musto et al. (2014), who examined four different ways to compute the overall polarity scores of microblogs: *basic*, *normalized*, *emphasized*, and *normalized-emphasized*. In each of these methods, the authors first split the input message into a list of *micro-phrases* based on the occurrence of punctuation marks and conjunctions. Afterwards, they calculated the polarity score for each of these segments and finally estimated the overall polarity of the whole tweet by uniting the scores of its micro-phrases. Musto et al. obtained their best results (58.99% accuracy on the SemEval-2013 dataset) with the normalized-emphasized approach, in which they averaged the polarity scores of segments' tokens, boosting these values by 50% for informative parts of speech (adjectives, nouns, and adverbs), and computed the final overall polarity of the microblog by taking the sum of all micro-phrase scores.

Another Twitter-aware system was presented by Jurek et al. (2015) who computed the negative and positive polarity of a message (F_p and F_n respectively) using the following formula:

$$F_P = \min\left(\frac{A_P}{2 - \log(3.5 \times W_P + I_P)}, 100\right), \tag{6.1}$$

$$F_N = \max\left(\frac{A_N}{2 - \log(3.5 \times W_N + I_N)}, -100\right); \tag{6.2}$$

where A_P and A_N represent the average scores of positive and negative lexicon terms found in the tweet; W_P and W_N stand for the raw counts of polar tokens; and I_P and I_N denote the number of intensifiers preceding these words. In addition to that, before estimating the average values, the authors also modified the polarity scores s_w every negated word w using the following heuristics:

$$neg(s_w) = \begin{cases} \min\left(\frac{s_w - 100}{2}, -10\right) & \text{if } s_w > 0, \\ \max\left(\frac{s_w + 100}{2}, 10\right), & \text{if } s_w < 0. \end{cases} \tag{6.3}$$

Furthermore, besides computing the polarity scores F_p and F_n, Jurek et al. also determined the subjectivity degree of the message by replacing the A_P and A_N terms in Equation 6.2 with the average of conditional probabilities of the tweet being subjective given the occurrences of the respective polar terms.[4] The authors considered a microblog as neutral if its absolute polarity was less than 25 and the subjectivity value was not greater than 0.5. Otherwise, they assigned a positive or negative label to this message depending on the sign of the polarity score. With this approach, Jurek et al. achieved an accuracy of 77.3% on the manually annotated subset of Go et al.'s corpus and reached 74.2% on the IMDB review corpus (Maas et al., 2011).

Finally, Kolchyna et al. (2015) also explored two different ways of computing the overall polarity of a microblog: (i) by simply averaging the scores of the lexicon terms found in the

[4]These probabilities were calculated automatically on the noisily labeled data set of Go et al. (2009).

message and (ii) by taking the signed logarithm of this average:

$$\text{Score}_{\log} = \begin{cases} \text{sign}(\text{Score}_{\text{AVG}}) \log_{10}(|\text{Score}_{\text{AVG}}|) & \text{if } |\text{Score}_{\text{AVG}}| > 0.1, \\ 0, & \text{otherwise;} \end{cases}$$

comparing theses approaches on the SemEval-2013 data (Nakov et al., 2013). The authors determined the final polarity of a tweet using the k-means clustering, which utilized both of the above polarity values as features. They showed that the logarithmic strategy performed better than the simple average solution, yielding an accuracy of 61.74%.

As it was unclear how each of these methods would perform on PotTS and SB10k, we reimplemented the approaches of Hu and Liu (2004) (as a relatively simple baseline), Taboada et al. (2011), Musto et al. (2014), Jurek et al. (2015), and Kolchyna et al. (2015), and applied these systems to the test sets of these corpora.

Following our comparison in Chapter 4, we chose the Zurich Polarity List as the primary sentiment lexicon for the tested methods. However, a significant drawback of this resource, which unfortunately slipped through our previous intrinsic evaluation, is that most of its entries have uniform weights, with their polarity scores being either 0.7 or 1. We decided to keep the original values as is, and only multiplied the scores of negative terms by -1, since all of the tested approaches presupposed different signs for the terms with opposite semantic orientations.[5] Moreover, because some analyzers (e.g., Taboada et al. (2011) and Musto et al. (2014)) relied on the part-of-speech tags of lexicon entries, we automatically tagged all terms in the polarity list with the help of the `TreeTagger` (Schmid, 1995), choosing the most probable part-of-speech tags for each entry and also using the tag sequences whose probabilities were at least two times lower than the likelihood of the best tagging assignment, duplicating the lexicon terms in the second case.

Furthermore, since all of the systems except for that of Kolchyna et al. (2015) by default returned continuous real values, but our evaluation required discrete polarity labels (*positive*, *negative*, or *neutral*), we discretized the results of these approaches using the following simple procedure: We first determined the optimal threshold values for each particular polar class on the training and development sets,[6] and then derived polarity labels for the test messages by comparing their predicted SO scores with these thresholds. To achieve the former goal (i.e., find the optimal thresholds), we exhaustively searched through all unique polarity values assigned to the training and development instances and checked whether using these values as a boundary between two adjacent polarity classes (sorted in ascending order of their positivity) would increase the overall macro-F_1 on the train and dev sets.

The final results of this evaluation are shown in Table 6.2.

[5] We will investigate the impact of other lexicons with presumably better scoring later in Section 6.6.2.

[6] Since none of the methods required training or involved any sophisticated hyper-parameters, we used both training and development data to optimize the threshold scores.

Method	Positive			Negative			Neutral			Macro	Micro
	Precision	Recall	F_1	Precision	Recall	F_1	Precision	Recall	F_1	$F_1^{+/-}$	F_1
PotTS											
HL	0.75	**0.76**	**0.76**	0.53	0.43	0.47	0.67	0.73	0.69	**0.615**	**0.685**
TBD	**0.77**	0.71	0.74	**0.54**	0.39	0.45	0.63	0.77	0.69	0.597	0.674
MST	0.75	0.72	0.74	0.48	**0.47**	**0.48**	**0.68**	0.72	0.7	0.606	0.675
JRK	0.6	0.31	0.41	0.42	0.2	0.27	0.43	0.8	0.56	0.339	0.467
KLCH	0.71	0.72	0.71	0.34	0.17	0.22	0.66	**0.82**	**0.73**	0.468	0.651
SB10k											
HL	**0.49**	**0.62**	**0.55**	0.27	0.33	0.3	**0.73**	0.62	0.67	**0.421**	0.577
TBD	0.48	0.6	0.53	0.24	0.27	0.25	0.72	0.63	0.67	0.393	0.57
MST	0.45	0.49	0.47	0.29	**0.35**	**0.32**	0.7	0.64	0.67	0.395	0.568
JRK	0.41	0.39	0.4	**0.36**	0.26	0.3	0.69	0.75	0.72	0.351	0.592
KLCH	0.39	0.22	0.28	0.34	0.13	0.19	0.66	**0.86**	**0.75**	0.235	**0.606**

Table 6.2: Evaluation of lexicon-based CGSA methods
HL – Hu and Liu (2004), TBD – Taboada et al. (2011), MST – Musto et al. (2014), JRK – Jurek et al. (2015), KLCH – Kolchyna et al. (2015)

As we can see, the performance of the tested methods significantly varies across different polarity classes, but follows more or less the same pattern on both datasets: For example, the most simple approach of Hu and Liu (2004) achieves surprisingly good quality at predicting positive tweets, showing the highest recall and F_1-measure on the PotTS corpus and yielding the best overall scores for this polarity class on the SB10k set. Moreover, on the latter data, it also outperforms all other systems with respect to the precision of neutral microblogs. Combined with its generally good results on other metrics, this classifier attains the highest macro-averaged F_1-result for all classes and sets up a new benchmark for the micro-F_1 on the PotTS test set.

The approach of Taboada et al. (2011), which can be viewed as an extension of the previous method, only surpasses the HL classifier in terms of the precision of positive and negative messages, but still loses more than 0.02 macro-F_1 due to a lower recall of the neutral class. A better performance in this regard is shown by the analyzer of Musto et al. (2014), which shows a fairly strong recall of negative tweets, which in turn leads to the best F_1-score for this polarity. Unfortunately, since this semantic orientation is the most underrepresented one in both corpora, this success is not reflected in the overall statistics: Although this methods ranks second in terms of the macro-averaged F_1, it lags behind its competitors with regard to the micro-averaged value on the SB10k corpus.

Finally, the system of Kolchyna et al. (2015) shows very strong recall and F_1-scores for the neutral class on both sets and also achieves the best accuracy (0.606) on the SB10k data, but its quality for the remaining two polarities (positive and negative) is fairly suboptimal, with the F_1-scores for these semantic orientations ranking last or second to last in both cases.

6.3.1 Polarity-Changing Factors

Since the analysis of context factors is commonly considered to be one of the most important components of any lexicon-based CGSA system, and because the method with the simplest approach to this task achieved surprisingly good results, outperforming other more sophisticated competitors, we decided to recheck the utility of this module for all classifiers. In order to do so, we successively deactivated, one by one, parts of the classifiers which analyzed the surrounding context of the polar terms and recomputed the F_1-scores of all systems after these changes.

Polarity-Changing Factors	**System Scores**									
	HL		**TBD**		**MST**		**JRK**		**KLCH**	
	Macro $F_1{}^{+/-}$	Micro F_1	Macro $F_1{}^{+/-}$	Micro F_1	Macro $F_1{}^{+/-}$	Micro F_1	Macro $F_1{}^{+/-}$	Micro F_1	Macro $F_1{}^{+/-}$	Micro F_1
PotTS										
All	0.615	0.685	0.593	0.671	0.606	0.675	0.339	0.467	0.468	0.651
–Negation	0.622	**0.691**	0.596	0.672	**0.641**	0.7	0.357	0.473	0.298	0.463
–Intensification	NA	NA	0.595	0.672	NA	NA	0.339	0.467	NA	NA
–Other Modifiers	NA	NA	0.613	0.684	NA	NA	NA	NA	NA	NA
SB10k										
All	**0.421**	0.577	0.392	0.569	0.395	0.568	0.351	0.592	0.235	0.606
–Negation	0.415	0.576	0.395	0.572	0.381	0.559	0.316	0.586	0.218	**0.609**
–Intensification	NA	NA	0.4	0.576	NA	NA	0.352	0.59	NA	NA
–Other Modifiers	NA	NA	0.406	0.566	NA	NA	NA	NA	NA	NA

Table 6.3: Evaluation of polarity-changing factors in lexicon-based CGSA methods HL – Hu and Liu (2004), TBD – Taboada et al. (2011), MST – Musto et al. (2014), JRK – Jurek et al. (2015), KLCH – Kolchyna et al. (2015)

As we can see from the results in Table 6.3, various methods respond in different ways to this ablation: For example, the scores of the Hu and Liu system improve on the PotTS corpus, but degrade on the SB10k dataset after switching off the negation handling. The same situation can also be observed with the analyzers of Musto et al. (2014) and Jurek et al. (2015). The classifier of Taboada et al. (2011), however, benefits from this deactivation in both cases, and the approach of Kolchyna et al. (2015) vice versa shows a performance drop on either dataset with the only exception being the micro-averaged F_1 on the SB10k data, which unexpectedly improves from 0.606 to 0.609.

As to the intensification handling, we can see that only two approaches (TBD and JRK) feature this component at all. Similarly to the previous case, the Taboada system profits from its deactivation, with the macro- and micro-averaged F_1-scores going up by $\approx 2e^{-3}$ on PotTS and $\approx 8e^{-3}$ on the SB10k corpus. A more varied situation is observed with the analyzer of Jurek et al. (2015), whose PotTS results are virtually unaffected by these changes, but the macro-averaged F_1 slightly increases and the micro-averaged score slightly decreases

on the Cieliebak et al.'s dataset.

Finally, "other modifiers" (such as irrealis and interrogative clauses) only play a role as a polarity-changing factor in the system of Taboada et al. (2011) and, as we can see from the figures, do there rather more harm than good: deactivating this part boosts the macro-averaged F_1-scores on PotTS and SB10k by $2e^{-2}$ and $14e^{-3}$ respectively. At the same time, the micro-averaged result of this system climbs up from 0.671 to 0.684 on the former dataset, but drops from 0.569 to 0.566 on the latter corpus.

6.3.2 Error Analysis

In order to get a better intuition about the strengths and weaknesses of each particular classifier and understand the way these methods work, we additionally collected a set of errors which were specific to only one of above the systems and will discuss some of these cases here in detail.

The first such error, which has been made by the system of Taboada et al. (2011), is shown in Example 6.3.1. Here, a strongly positive tweet describing one's excitement about a technical report has been erroneously classified as neutral despite the presence of the prototypical positive term "gut" (*good*) in its superlative form "beste" (*best*). Unfortunately, it is the degree of comparison which becomes fatal in this case: According to the implementation of Taboada et al. (2011), any superlative adjective has to be preceded by the definite article and a verb in order to be considered as a polar term for the final SO computation. Although the adjective "beste" (*best*) can fulfill the first criterion (it immediately follows the determiner "der" [*the*]), the lack of the preceding verb nullifies its effect.

Example 6.3.1 (Error Made by the System of Taboada et al.)
Tweet: Der beste Microsoft Knowledgebase-Artikel, den ich je gelesen habe.
The best Microsoft-Knowledgebase article I've ever read.

Gold Label: positive
Predicted Label: **neutral***

Another error of this method is shown in Example 6.3.2. This time, the presence of the colloquial term "verarschen" (*to hoax*) suggests that the tweet at hand is negative. Alas, the occurrence of another verb ("wollt" [*wanna*]) is interpreted as an irrealis clue, which prevents further SO computation and leads to a zero score to the whole message.

Example 6.3.2 (Error Made by the System of Taboada et al.)
Tweet: Die Konklave wählt den Papst und dann sagen sie Gott war es - Wollt ihr mich verarschen ?!
The conclave elects the Pope and then they say it was God - do you wanna hoax me?!

Gold Label: **negative**
Predicted Label: **neutral***

At this point, we already can see that the main flaws of the TBD approach apparently stem from its overly coarse rules, which, in addition, are not always valid in the Twitter domain.[7] In this regard, we can easily imagine that applying this classifier to longer and more formal texts such as newspapers or journal articles would yield significantly better results as the analyzer would not depend on just one occurrence of a polar term in the input and its rules would presumably be better suited for better formed data.

Returning back to our error analysis, let us look at another erroneous case shown in Example 6.3.3. This time, the system of Musto et al. (2014) has incorrectly assigned the neutral label to a positive tweet even though the positive term "gut" (*good*) again appears in this message. As it turns out, the occurrence of this word is still insufficient for the classifier to predict the positive class although this term has the highest possible positive score in the lexicon (1.0), which is additionally boosted by a factor of 1.5, since the word belongs to an informative part of speech. However, the crushing factor in this case is the length of the tweet: since this approach relies on the average SO-score for all words in a sentence, the value 1.5 of the only positive term gets divided by 7 (the length of the sentence) and drops down to 0.214, which is below the threshold for the positive class (0.267).

Example 6.3.3 (Error Made by the System of Musto et al.)
Tweet: Mensch Meier, Mensch Meier! Das sieht gut aus für die %User:
Gosh Meier, Gosh Meier! It looks good for the %User:

Gold Label: **positive**
Predicted Label: **neutral***

As it turns out, this kind of mistakes is by far the most common type of errors characteristic to the MST system. Further examples of such incorrect decisions are provided in Example 6.3.4:

[7] In order to check this claim, we tried to temporarily deactivate the above two heuristics (predicate check for superlative adjectives and irrealis blocking by model verbs) and recomputed the scores of this system, getting in both cases an improvement by almost one percent on either corpus.

Example 6.3.4 (Errors Made by the System of Musto et al.)
Tweet: Der %User tut echt geile musik machen. Nichts mit Boyband hier.
The %User is making really great music. Nothing with Boyband here.

Gold Label: positive
Predicted Label: neutral*

Tweet: Diese S5E5 Episode mit den Zugüberfall war wieder genial! BreakingBad
This S5E5 episode with train robbery was brilliant again! BreakingBad

Gold Label: positive
Predicted Label: neutral*

A different kind of problems is experienced by the approach of Jurek et al. (2015), which apparently has difficulties with correctly predicting the positive class. A deeper analysis of its misclassifications revealed that the reason for it is relatively simple: As you might remember from the initial description of this system, this classifier is the only method which uses conditional probabilities of polar terms instead of their lexicon values. We estimated these probabilities on the noisily labeled German Twitter Snapshot, which unfortunately was extremely biased towards the positive class (see Table 6.1). As a consequence of this, positive lexicon entries got extremely high likelihood values, so that even a single occurrence of one such term outweighed the effect of multiple negative expressions. This is, for instance, the case in Example 6.3.5 where the score of the (questionable) positive expression "Ok" is greater than the absolute sum of two negative values for the terms "sich beschweren" (*to complain*) and "ekelhaft" (*disgusting*).

Example 6.3.5 (Error Made by the System of Jurek et al.)
Tweet: Normal bin ich ja nicht der mensch dwer sich beschwert wegen dem essen aber diese Pizza von Joeys... boah wie ekelhaft
Ok I'm not a person who complains about food but this pizza from Joeys... Damn it's so disgusting

Gold Label: negative
Predicted Label: positive*

The same problem also afflicts the system of Kolchyna et al. whose error example is given in 6.3.6. In contrast to the previous approaches, which mainly rely on manually designed heuristic rules, this method makes its decisions using a trained k-NN classifier. Nevertheless,

its prediction in the provided case is still incorrect as it evidently confuses the positive class with the neutral polarity.

Example 6.3.6 (Error Made by the System of Kolchyna et al.)
Tweet: das Hört sich echt Super an! %PosSmiley macht sami nicht auch so ein Video? Noah süsse beste Freunde! ♡%User isilie saminator
It sounds really fantastic! %PosSmiley won't sami also make such a video? Noah's sweet best friends! ♡%User isilie saminator

Gold Label: positive
Predicted Label: neutral*

In order to better understand the reason for this misclassification, we first looked at the initial SO scores computed by the Kolchyna analyzer (as you might remember, the internal k-NN predictor of this system uses the average SO value of all polar terms found in the message and the logarithm of this value as its features). As it turned out, both of these scores (mean SO and its log) were relatively high, amounting to 33.42 and 2.52 respectively. However, a closer look at the selected nearest neighbors revealed that, even despite such high SO values, top three of the closest neighbors of this microblog were in fact neutral. You can see the full list of the five nearest neighbors below:

1. **Tweet:**: "Not in my backyard" -Mentalität dt. Politik: "Nächster Castor geht wohl doch nach Gorleben... -%Link antiatom"
   "Not in my backyard" -Mentality of German politics: "Next Castor will probably still got to Gorleben... -%Link antiatom"
   Label: neutral
   Distance: $6.83e^{-03}$;

2. **Tweet:**: Kanzlerin im Google-Hangout: "Die Technik soll sich mal bemühen"
   Chancellor in Google-Hangout: "The technology should make an effort"
   Label: neutral
   Distance: $1.6e^{-02}$;

3. **Tweet:**: Kanzlerin im Google-Hangout: "Die Technik soll sich mal bemühen"
   Chancellor in Google-Hangout: "The technology should make an effort"
   Label: neutral
   Distance: $1.6e^{-02}$;

4. **Tweet:**: Wünsche mir ein Format wie zdflogin auch für das %User. Viele Themen, klare Aussagen. Schönes Special %User zur Landtagswahl! %PosSmiley

Wish %User had a format like zdflogin. Many topics, clear statements. Nice Special %User zur Landtagswahl! %PosSmiley
Label: positive
Distance: $2.1e^{-02}$;

5. **Tweet:**: Ich bin ja so gespannt ob die FDP im September erst den Zahnärzten und dann den Apothekern mit Geschenken dankt, oder anders rum...
I'm so curious whether FDP will first give gifts to dentists and then to pharmacists in September, or whether it'll be vice versa
Label: positive
Distance: $4.12e^{-02}$;

Even more surprisingly, the SO scores of the neighboring neutral instances were indeed also relatively high: In the first microblog, for example, the system recognized two polar terms: "Not" (*Not*) which was confused with "Not" (*distress*) and "nächster" (*next*). Another polar expression ("sich bemühen" [*to make an effort*]) was found in messages 2 and 3. Although two of these terms ("Not" and "sich bemühen") had a negative label in the sentiment lexicon, their conditional probability of being associated with the positive class was more than ten times bigger than the chance to appear in a negative microblog (according to the computed statistics). As a consequnece of this positive probability bias, many of the neutral training tweets ended up in close vicinity to the actual positive examples.

As you can see, lexicon-based methods experience various kinds of problems with predicting the polarity of short casually written microblogs: Some of these systems apply rules which are too domain-specific and do not generalize well to Twitter posts, others rely on noisy statistics which might be extraordinarily skewed towards just one polarity. Now, we should check whether other approaches to the coarse-grained sentiment analysis (which rely on completely different principles and paradigms) will also be susceptible to these kinds of errors.

6.4 Machine-Learning Methods

Despite their immense popularity, linguistic plausibility, and simplicity to implement, lexicon-based approaches often have been criticized for the rigidness of their classification[8] and the inability to incorporate additional, non-lexical attributes into their final decisions. Moreover,

[8]Since these systems only rely on the precomputed weights of lexicon entries, considering these coefficients as constant, their decision boundaries frequently appear to be suboptimal as many terms might have different polarity and intensity values depending on the domain (see Eisenstein, 2017; Yang and Eisenstein, 2017).

as noted by Pang et al. (2002) and also confirmed empirically by Riloff et al. (2003) and Gamon (2004), many linguistic expressions which actually correlate with the subjectivity and polarity of a sentence (e.g., exclamation marks or spelling variations) are very unlikely to be included into a sentiment lexicon even by a human expert. As a consequence of this, with the emergence of manually annotated corpora, lexicon-based systems have been gradually superseded by supervised machine-learning methods.

One of the first steps in this direction was taken by Wiebe et al. (1999) who used a Naïve Bayes classifier to differentiate between subjective and objective statements. Using primarily binary features which reflected the presence of a pronoun, an adjective, a cardinal number, or a modal verb in the analyzed sentence, the authors achieved an accuracy of 72.17% on the two-class prediction task (differentiating between positive and negative classes), outperforming the majority class baseline by more than 20%. An even better result (81.5%) could be reached when the dataset was restricted only to the examples with the most confident annotation.

Inspired by the success of this approach, Yu and Hatzivassiloglou (2003) presented a more elaborated system in which they first distinguished between subjective and objective documents, then differentiated between polar and neutral sentences, and, finally, determined the semantic orientation of opinionated clauses. As in the previous case, the authors used a Naïve Bayes predictor for the document-level task, reaching a remarkable F_1-score of 0.96 on this objective; and applied an ensemble of NB systems to predict the subjectivity of single sentences. To determine the semantic orientation of subjective clauses, Yu and Hatzivassiloglou averaged the polarity scores of their tokens, obtaining these scores from an automatically constructed sentiment lexicon (Hatzivassiloglou and McKeown, 1997). This way, they attained an accuracy of 91% on a set of 38 sentences that had a perfect inter-annotator agreement.

In order to check the effectiveness of the Naïve Bayes approach, Pang et al. (2002) compared the results of NB, MaxEnt, and SVM systems on the movie review classification task, trying to predict whether a review was perceived as thumbs up or thumbs down. In contrast to the previous works, they found the SVM classifier working best for this objective, yielding 82.9% accuracy when used with unigram features only. This conclusion paved the way for the following triumph of the support-vector approach, which dominated the whole CGSA research field for almost a decade ever since. For example, Gamon (2004) also trained an SVM predictor on a rich set of linguistic and surface-level features (including part-of-speech trigrams, context-free phrase-structure patterns, and part-of-speech information coupled with semantic relations) to distinguish between positive and negative customer feedback, achieving 77.5% accuracy and $\approx$0.77 F_1 by using only top 2,000 attributes that had the highest log-likelihood ratio with the target class. Furthermore, Pang and Lee (2005) addressed

the problem of multi-class rating, attempting to predict the number of stars that an author could assign to a review. For this purpose, they compared three different SVM types: (i) one-versus-all SVM (OVA-SVM), (ii) SVM regression, (iii) and OVA-SVM with metric labeling; getting their best results (≈52% accuracy) with the last option. Finally, Ng et al. (2006) proposed another multi-stage SVM system, in which they first classified whether the given text was a review or not and then tried to predict its polarity. Due to a better usage of higher-order n-grams (where, instead of naïvely considering all token sequences up to length n as new features, the authors only took 5,000 most useful ones, measuring their utility with the weighted log-likelihood ratio), Ng et al. (2006) even improved the state of the art on the Pang and Lee's corpus, boosting the classification accuracy from 87.1 to 90.5%.

However, a real game change in the CGSA research field happened with the introduction of the SemEval shared task on sentiment analysis in Twitter (Nakov et al., 2013). Starting from its inaugural run in 2013, this competition has rapidly caught the attention of the broader NLP community and has been rerun five times, attracting more than 40 active participants every year.

It is not surprising that the first winning systems in this task closely followed in the footsteps of the advances in the general opinion mining at that time. For example, the two top-scoring submissions in the initial iteration (Mohammad et al., 2013; Günther and Furrer, 2013) both relied on the SVM algorithm: The first of these approaches—an analyzer developed by Mohammad et al. (2013)—was the absolute winner of SemEval 2013, scoring impressive 0.69 macro-averaged two-class F_1 on the provided Twitter corpus. The key to the success of this method was an extensive set of linguistic features devised by the authors, which included character and token n-grams, Brown clusters (Brown et al., 1992), statistics on part-of-speech tags, punctuation marks, elongated words etc. However, the most useful type of attributes according to the feature ablation test turned out to be the features that reflected information from various sentiment lexicons. In particular, depending on the type of the polarity list from which such information was extracted, Mohammad et al. introduced two types of lexicon attributes: *manual* and *automatic* ones. The former group was computed with the help of the NRC emotion lexicon (Mohammad and Turney, 2010), MPQA polarity list (Wilson et al., 2005), and Bing Liu's manually compiled polarity set (Hu and Liu, 2004). For each of these resources and for each of the non-neutral polarity classes (positive and negative), the authors estimated the total sum of the lexicon scores for all message tokens and also separately calculated these statistics for each particular part-of-speech tag, considering them as additional attributes. Automatic features were obtained using the Sentiment140 and Hashtag Sentiment Base polarity lists (Kiritchenko et al., 2014). Again, for each of these lexicons, for each of the two polarity classes, the authors produced four features representing the number of tokens with non-zero scores, the sum and the maximum of all respective lexicon values for all words, and the score of the last term in the tweet. These two feature

groups (manual and automatic lexicon attributes) improved the macro-averaged $F_1{}^{+/-}$-score by almost five percent, outperforming in this regard all other traits.

Another notable submission—the system of Günther and Furrer (2013)—also relied on a linear SVM predictor with a rich set of features. Similarly to Mohammad et al. (2013), the authors used unmodified and lemmatized unigrams, word clusters, and lexicon features. However, in contrast to the previous approach, this application utilized only one polarity list—that of Esuli and Sebastiani (2005). Partially due to this fact, Günther and Furrer found the word clusters working best among all features. This method also yielded competitive results (0.653 $F_1{}^{+/-}$) on the message-level polarity task, attaining second place in that year.

Later on, Günther et al. (2014) further improved their results (from 0.653 to 0.691 two-class F_1) by extending the original system with a Twitter-aware tokenizer (Owoputi et al., 2013), spelling normalization module, and a significantly increased set of lexicon-based features: In particular, instead of simply relying on SentiWordNet (Esuli and Sebastiani, 2005), Günther et al. applied a whole ensemble of various polarity lists including Liu's opinion lexicon, MPQA subjectivity list, and TwittrAttr polarity resource. As mentioned by the authors, the last change was of particular use to the classification accuracy, improving the macro-$F_1{}^{+/-}$ by almost four percent.

An even better score on this task, could be attained with the approach of Miura et al. (2014) who also utilized a supervised ML classifier with character and word n-grams, word clusters, disambiguated senses, and lexicon scores of message tokens as features. Similarly to the systems of Mohammad et al. (2013) and Günther et al. (2014), the authors made heavy use of various kinds of polarity lists including AFINN-111 (Nielsen, 2011), Liu's Opinion Lexicon (Hu and Liu, 2004) , General Inquirer (Stone et al., 1966), MPQA Polarity List (Wiebe and Riloff, 2005), NRC Hashtag and Sentiment140 Lexicon (Mohammad et al., 2013), as well as SentiWordNet (Esuli and Sebastiani, 2006c), additionally applying a whole set of preprocessing steps such as spelling correction, part-of-speech tagging with lemmatization, and a special weighting scheme for underrepresented classes. Due to these enhancements, combined with a carefully tuned LogLinear classifier, Miura et al. (2014) were able to boost the sentiment classification results on the SemEval 2014 test set to 0.71 $F_1{}^{+/-}$.

In order to see how this family of methods would perform on our Twitter corpora, we have reimplemented the approaches of Gamon (2004), Mohammad et al. (2013), and Günther et al. (2014) with the following modifications: For the feature extraction in the system of Gamon (2004), we used the available dependency analyses from the `MateParser` (Bohnet, 2009) instead of constituency trees, considering each node of the dependency tree as a syntactic constituent and regarding the two-tuple (`dependency-link-to-the-parent`, `PoS-tag-of-the-node`) as the name of that constituent (for example, a finite verb at the root node of the tree was mapped to the constituent (`-`, `VVFIN`), where `-` was the name of the root relation). Fur-

thermore, because the Brown clusters were not available for German, we had to remove this attribute altogether from the feature sets of Mohammad et al.'s and Günther et al.'s methods. However, because the former system relied on two types of lexicon attributes—manual and automatic ones, we used two polarity lists for these approaches—the Zurich Sentiment Lexicon of Clematide and Klenner (2010) as a manual resource and the Linear Projection Lexicon as an automatically generated polarity set. All remaining attributes and training specifics were kept maximally close to their original descriptions.

Method	Positive			Negative			Neutral			Macro	Micro
	Precision	Recall	F_1	Precision	Recall	F_1	Precision	Recall	F_1	F_1	F_1
PotTS											
GMN	0.67	0.73	0.7	0.35	0.15	0.21	0.6	0.72	0.66	0.453	0.617
MHM	**0.79**	0.77	**0.78**	**0.58**	**0.56**	**0.57**	**0.73**	**0.76**	**0.74**	**0.674**	**0.727**
GNT	0.71	**0.8**	0.75	0.55	0.45	0.5	0.68	0.63	0.65	0.624	0.673
SB10k											
GMN	0.65	0.45	0.53	0.38	0.08	0.13	0.72	**0.93**	0.81	0.329	0.699
MHM	**0.71**	**0.65**	**0.68**	**0.51**	**0.4**	**0.45**	**0.8**	0.87	**0.84**	**0.564**	**0.752**
GNT	0.67	0.62	0.64	0.44	0.28	0.34	0.78	0.87	0.82	0.491	0.724

Table 6.4: Evaluation of ML-based CGSA methods
GMN – Gamon (2004), MHM – Mohammad et al. (2013), GNT – Günther et al. (2014)

The results of our reimplementations are shown in Table 6.4. As we can see from the scores, the system of Mohammad et al. (2013) clearly dominates its competitors on both corpora. This holds for all presented metrics except for the recall of positive tweets on the PotTS dataset and neutral messages on the SB10k data, where it is outperformed by the analyzers of Günther et al. (2014) and Gamon (2004) respectively. In any other respect, however, the results of the MHM classifier are notably higher than those of the GNT method, sometimes surpassing it by up to 12 percent (this is, for instance, the case for the recall of negative microblogs on the SB10k corpus). This margin becomes even larger if we compare the scores of Mohammad's system with the performance of Gamon's predictor, which is by far the weakest ML method in our survey. This weakness, however, is less surprising regarding the fact that Gamon's approach is purely grammar-based and relies only on information about part-of-speech tags and constituency parses without any lexicon traits or even plain n-gram features. Partially due to these limited input attributes, the results of this analyzer are even worse than the average scores of lexicon-based methods.

6.4.1 Feature Analysis

Because input features appeared to play a crucial role for the success of ML-based systems, we decided to investigate the impact of this factor in more detail and performed an ablation

test for each of the tested classifiers, removing one of their feature groups at a time and recomputing their scores.

Features	System Scores GMN		MHM		GNT	
	Macro $F_1^{+/-}$	Micro F_1	Macro $F_1^{+/-}$	Micro F_1	Macro $F_1^{+/-}$	Micro F_1
PotTS						
All	**0.453**	**0.617**	**0.674**	0.727	**0.624**	0.673
–Constituents	0.388	0.545	NA	NA	NA	NA
–PoS Tags	0.417	0.607	0.669	0.721	NA	NA
–Character Features	NA	NA	0.671	**0.734**	NA	NA
–Token Features	NA	NA	0.659	0.704	0.0	0.366
–Automatic Lexicons	NA	NA	0.667	0.717	0.613	0.666
–Manual Lexicons	NA	NA	0.665	0.715	0.617	**0.675**
SB10k						
All	**0.329**	0.699	0.564	0.752	0.491	0.724
–Constituents	0.127	0.646	NA	NA	NA	NA
–PoS Tags	0.301	**0.7**	**0.57**	**0.757**	NA	NA
–Character Features	NA	NA	0.546	0.753	NA	NA
–Token Features	NA	NA	0.559	0.741	0.046	0.62
–Automatic Lexicons	NA	NA	0.54	0.753	**0.517**	0.735
–Manual Lexicons	NA	NA	0.553	0.751	0.51	**0.739**

Table 6.5: Results of the feature-ablation test for ML-based CGSA methods GMN – Gamon (2004), MHM – Mohammad et al. (2013), GNT – Günther et al. (2014)

As we can see from the results in Table 6.5, the approach of Gamon (2004) typically achieves its best performance when all of the input attributes (PoS tags and syntactic constituents) are active. This is for example the case for the micro- and macro-averaged F_1 on the PotTS corpus, and also holds for the two-class macro-F_1 on the SB10k data. The only exception to this tendency is the micro-averaged F_1-score on the latter dataset, which shows a slight improvement (from 0.699 to 0.7) after the removal of part-of-speech features.

Similarly, the analyzer of Mohammad et al. (2013) seems to rather suffer than benefit from the part-of-speech attributes, which decrease its micro-averaged scores by almost $7e^{-2}$ on PotTS and $5e^{-2}$ F_1 on SB10k. One possible explanation for this degradation could be the differences in the utilized PoS taggers and tagsets: Whereas the original Mohammad et al.'s classifier relied on a special Twitter-aware tagger (Owoputi et al., 2013), whose tags were explicitly adjusted to the peculiarities of social media texts (including special labels for the @-mentions and #hashtags), we instead used the output of the standard `TreeTagger` (Schmid, 1995), which, apart from lacking any Twitter-specific information, was also trained on a completely different text genre (newspaper articles) and therefore a priori produced unreliable output. As a consequence, the effect of the part-of-speech information is rather harmful, and the only aspect where it comes in handy is the macro-averaged F_1 on the PotTS corpus, which improves by $3e^{-3}$ when these features are used.

An even more controversial situation is observed with the analyzer of Günther et al. (2014). Although this system lacks any part-of-speech attributes, its reaction to the deletion of other features (first of all token and lexicon traits) is quite unexpected. For example, the macro-averaged F_1-scores on both corpora drop almost to zero when the information about tokens is excluded. On the other hand, the deactivation of manual lexicons surprisingly improves the micro-averaged results on both datasets and also increases the macro-${F_1}^{+/-}$ on the SB10k data. We also notice a similar (though less pronounced) tendency with the automatic lexicons: the ablation of these features lowers the scores on PotTS, but improves both results on SB10k. We can partially explain this negative effect of polarity lists by the coarseness of the lexicon features: The classifier uses only binary attributes reflecting whether the given tweet has more positive or more negative lexicon items, but does not distinguish between the scores or intensities of these terms.

Besides analyzing the utility of each particular feature group, we also decided to have a look at the top-10 most relevant attributes learned by each system. The summarized overview in Table 6.6 partially confirms our previous findings: For example, the most useful traits for the analyzer of Gamon (2004) are attributes reflecting the information about both constituents and part-of-speech tags, with five of its ten entries featuring the interjection tag, which appears to be especially important for predicting the positive class. On the other hand, the system of Mohammad et al. (2013) seems to rely more on token and character n-grams, as nine out of ten attributes belong to either of these two categories. The only outlier in this respect is the `Last%QMarkCnt` attribute (line 2), which denotes the presence of a question mark and is apparently a good clue of neutral microblogs. Finally, the classifier of Günther et al. (2014) almost exclusively prefers lexical n-grams, as it has nine unigrams and one bigram among its top-ten entries.

6.4.2 Effect of the Classifiers

Another important factor which could significantly affect the quality of ML-based approaches was the underlying classification method that was used to optimize the feature weights and make the final predictions. Although most of the previous studies agree on the superior performance of support vector machines for this task (see Pang et al., 2002; Gamon, 2004; Mohammad et al., 2013), we decided to question these conclusions as well and reran our experiments, replacing the linear SVC predictor with the Naïve Bayes and Logistic Regression algorithms.

Somewhat surprisingly, these changes indeed resulted in an improvement, especially in the case of the logistic classifier, which yielded the best macro- and micro-averaged scores for the systems of Mohammad et al. (2013) and Günther et al. (2014) on the PotTS corpus

Rank	GMN			MHM			GNT		
	Feature	Label	Weight	Feature	Label	Weight	Feature	Label	Weight
1	NK-ITJ|	POS	0.457	*	NEUT	0.131	hate	NEG	1.86
2	DM-ITJ|	POS	0.334	Last-%QMark-Cnt	NEUT	0.088	sick	NEG	1.7
3	V-DM-I	POS	0.244	s-c	NEG	0.079	kahretsinn	NEG	1.69
4	N-NK-I	POS	0.24	*-%possmiley	POS	0.067	dasisaberschade	NEG	1.69
5	MO-ITJ|	POS	0.211	c-h-e-i-s	NEG	0.064	Anziehen	POS	1.67
6	A-DM-I	POS	0.196	h-a-h	POS	0.064	\x016434	POS	1.65
7	A-MO-I	POS	0.191	t-_-.	NEG	0.064	pärchenabend	POS	1.65
8	NK-ITJ	POS	0.165	geil	POS	0.062	derien♡♡	POS	1.65
9	NK-$.	NEUT	0.16	*-?	NEUT	0.062	schön-nicht	POS	1.56
10	DM-ITJ	POS	0.157	?	NEUT	0.061	applause	POS	1.5

Table 6.6: Top-10 features learned by ML-based CGSA methods (sorted by the absolute values of their weights)

(see Table 6.7) and also produced the highest micro-F_1 results for these two approaches on the SB10k dataset. Nevertheless, the SVM algorithm still remains a competitive option, in particular for the feature-sparse method of Gamon (2004), but also with respect to the macro-F_1 of Mohammmad's and Günther's analyzers.

Features	System Scores					
	GMN		MHM		GNT	
	Macro $F_1^{+/-}$	Micro F_1	Macro $F_1^{+/-}$	Micro F_1	Macro $F_1^{+/-}$	Micro F_1
PotTS						
SVM	**0.453**	**0.617**	0.674	0.727	**0.624**	0.673
Naïve Bayes	0.432	0.577	0.635	0.675	0.567	0.59
Logistic Regression	0.431	0.612	**0.677**	**0.741**	**0.624**	**0.688**
SB10k						
SVM	0.329	**0.699**	**0.564**	0.752	0.491	0.724
Naïve Bayes	**0.351**	0.637	0.516	0.755	0.453	0.675
Logistic Regression	0.309	0.693	0.553	**0.772**	**0.512**	**0.75**

Table 6.7: Feature-ablation test of ML-based CGSA methods GMN – Gamon (2004), MHM – Mohammad et al. (2013), GNT – Günther et al. (2014)

Even though our results contradict previous claims in the literature, we would advise against premature generalizations and conclusions at this point and stress the fact that different classifiers might have fairly varying results on different datasets. Therefore, higher scores of the logistic regression on our corpora do not preclude better SVM results on the official SemEval data.

6.4.3 Error Analysis

Similarly to our experiments in the previous section, we also decided to have a closer look at the errors produced by each of the tested systems. For this purpose, we again collected misclassifications that were unique to only one of the classifiers, and provide some examples of these error cases below.

The first wrong result shown in Example 6.4.1 was produced by the system of Gamon (2004).

Example 6.4.1 (Error Made by the System of Gamon)
Tweet: Das ist das zynische. Über Themen labern, Leute schlecht machen. Wenn nicht der Papst damit Thema wäre, kein Wort. Ich hasse das.
It's cynical. To babble about topics, to talk people down. If the topic wouldn't be about the Pope, no word. I hate this.

Gold Label: negative
Predicted Label: positive*

In this case, the classifier incorrectly assigned a positive label to a clearly negative microblog despite the presence of multiple negatively connoted terms ("zynische" [*cynical*], "labern" [*to babble*], "schlecht machen" [*to talk down*], and "hasse" [*to hate*]). The reason for this decision was quite simple: As we already noted in the foregoing description, this method is completely unlexicalized and relies only on grammatical information while making its predictions. In particular, for this microblog, the top-5 most important features (ranked by the absolute values of their coefficients) were:

1. `PD-ADJA` (neutral): -0.62896911412,
2. `--VVINF` (negative): 0.517300341184,
3. `PD-ADJA` (positive): 0.505413668274,
4. `--VVINF` (positive): -0.346990702756,
5. `CJ-VVINF` (positive): 0.303311030403.

As you can see, none of these attributes reflects any information about the lexical terms appearing in the message, and the system simply prefers the positive class based on the presence of a predicate adjective (`PD-ADJA`) and coordinately conjoined infinitive (`CJ-VVINF`).

Another error shown in Example 6.4.2 was made by the system of Mohammad et al. (2013). This time, a positive tweet was misclassified as neutral. However, the reason for this

erroneous decision is completely different. As we can see from the list of the highest ranked features given below:

1. `*` (neutral): 0.131225868029,
2. `*` (negative): -0.0840804221845,
3. `%PoS-CARD` (neutral): 0.0833658576233,
4. `%PoS-ADJD` (neutral): -0.069745190018,
5. `t-␣-n` (positive): 0.0556721202587;

this analyzer makes its decision based on rather general, but excessively heavy weighted features (such a the placeholder token `*` or PoS-tag traits `%PoS-CARD` and `%PoS-ADJD`), and consequently succumbs to the neutral bias of these general traits.

Example 6.4.2 (Error Made by the System of Mohammad et al.)
Tweet: das klingt richtig gut! Was für eine hast du denn? (uvu) %PosSmiley3
It sounds really great. Which one do you have? (uvu) %PosSmiley3

Gold Label: **positive**
Predicted Label: **neutral***

Finally, the last example (6.4.3) shows another wrong decision, where a negative microblog was incorrectly analyzed as positive by the method of Günther et al. (2014), even though the polar term "borniert" (*narrow-minded*) was present in both utilized sentiment lexicons (ZPL and linear Projection) as a negative item. This again can be explained by the prevalence of general features (e.g., `8`, `nicht-nur_NEG`, `nur_NEG`, etc.) and their strong bias towards the majority class in the PotTS dataset.

Example 6.4.3 (Error Made by the System of Günther et al.)
Tweet: Den CDU-Wählern traue ich durchaus zu der FDP 8 bis 9% zu bescheren! Die sind so borniert, nicht nur in Niedersachsen!
I don't put giving 8 to 9% to the FDP past the CDU-voters! They are so narrow-minded, not only in Lower Saxony!

Gold Label: **negative**
Predicted Label: **positive**

6.5 Deep-Learning Methods

Even though traditional ML-based approaches still show competitive results and play an important role in the sentiment analysis of social media, they are gradually giving place to allegedly more powerful and in a certain sense more intuitive deep learning (DL) methods. As we already mentioned in the previous chapter, in contrast to the standard supervised techniques with human-engineered features, DL systems induce their input attributes completely automatically, and, in some cases, might produce an even better feature representation than the one devised by a human expert. Another important advantage of this paradigm is its more straightforward way to implement the "compositionality" of language (Frege, 1892): Whereas conventional classifiers usually consider each instance as a *bag of features* and predict the label based on the sum of these features' values multiplied with their respective scores, DL approaches try to *combine* the representation of each part of the instance (be it tokens or sentences) into a single whole and then deduce the final class from this joint embedding.

Among the first who explicitly incorporated the compositionality principle into a DL-based sentiment application were Yessenalina and Cardie (2011). In their proposed matrix-space approach, they represented each word w of an input phrase $\mathbf{x}^i = w_1^i, w_2^i, \ldots, w_{|\mathbf{x}^i|}^i$ as a matrix $W_w \in \mathbb{R}^{m \times m}$ and computed the sentiment score ξ^i of this phrase as the product of its token matrices, multiplying the final result with two vectors ($\vec{u}$ and $\vec{v} \in \mathbb{R}^m$) to get a scalar value:

$$\xi^i = \vec{u}^\top \left(\prod_{j=1}^{|x^i|} W_{w_j^i} \right) \vec{v}.$$

After computing this term, the authors predicted the intensity and polarity of the phrase on a five-level sentiment scale (ranging from very negative to very positive) by comparing ξ^i with automatically derived thresholds. With this system, Yessenalina and Cardie attained a ranking loss of 0.6375 on the MPQA corpus (Wiebe et al., 2005), outperforming the traditional PRank algorithm (Crammer and Singer, 2001) and bag-of-words ordered logistic regression.

Almost contemporarily with this work, Socher et al. (2011) introduced a deep recursive autoencoder (RAE), in which they obtained a fixed-width vector representation for a complex phrases $\vec{w}_p$ by recursively merging vectors of its tokens over a binarized dependency tree, first multiplying these vectors with a compositional matrix W and then applying a non-linear function ($softmax$) to the resulting product:

$$\vec{w}_p = softmax\left(W \begin{bmatrix} \vec{w}_l \\ \vec{w}_r \end{bmatrix} \right), \tag{6.4}$$

where $\vec{w}_l$ represents the vector of the left child and $\vec{w}_r$ denotes the embedding of the right dependent. By applying a max-margin classifier to the final phrase vector, the authors could improve the state-of-the-art results on predicting the sentence-level polarity of user's blog posts (Potts, 2010) and also outperformed the system of Nasukawa and Yi (2003) on the MPQA data set (Wiebe et al., 2005), achieving 86.4% accuracy on predicting contextual polarity of opinionated expressions.

Later on, Socher et al. (2012) further improved this approach by associating an additional matrix W_w with each vocabulary word w similarly to Yessenalina et al. (2010) and performing the inference simultaneously over both vector and matrix representations:

$$\vec{w}_p = tanh\left(W_v \begin{bmatrix} W_r\vec{w}_l \\ W_l\vec{w}_r \end{bmatrix}\right),$$

$$W_p = W_m \begin{bmatrix} W_l; \\ W_r \end{bmatrix};$$

where $\vec{w}_p \in \mathbb{R}^n$ stands for the vector embedding of the parent node, $\vec{w}_l$ and $\vec{w}_r$ represent the vectors of its left and right children, and $W_p, W_l, W_r \in \mathbb{R}^{n\times n}$ denote the respective matrices associated with these vertices. The compositionality matrices $W_v \in \mathbb{R}^{n\times 2n}$ and $W_m \in \mathbb{R}^{n\times 2n}$ were shared across all instances and learned along with the vector embeddings. This model, called Matrix-Vector Recursive Neural Network (MVRNN), surpassed the RAE system on the IMDB movie review dataset (Pang and Lee, 2005), attaining 0.91 Kullback-Leibler divergence between the assigned scores and probabilities of correct labels.

Yet another improvement—a Recursive Neural Tensor Network (RNTN)—was presented by Socher et al. (2013). In this system, the authors again opted for a vector representation of words, but enhanced the original matrix-vector product from Equation 6.4 with an additional tensor multiplication:

$$\vec{w}_p = softmax\left(\begin{bmatrix} \vec{w}_l \\ \vec{w}_r \end{bmatrix}^\top V^{[1:d]} \begin{bmatrix} \vec{w}_l \\ \vec{w}_r \end{bmatrix} + W \begin{bmatrix} \vec{w}_l \\ \vec{w}_r \end{bmatrix}\right),$$

where $\vec{w}_p, \vec{w}_l, \vec{w}_r \in \mathbb{R}^n$, and $W \in \mathbb{R}^{n\times 2n}$ are defined as previously and denote the word embeddings and compositionality matrix respectively; and V represents a $2n \times 2n \times n$-dimensional tensor. By increasing this way the number of parameters in comparison with the RAE approach, but significantly reducing it with respect to the MVRNN method, the authors gained a significant improvement of the results, boosting the classification accuracy on their own Stanford Sentiment Treebank from 82.9 to 85.4%.

A real breakthrough in the use of deep learning methods for sentiment analysis of Twitter happened with the work Severyn and Moschitti (2015b), whose proposed feed-forward DL system ranked first in SemEval-2015 Subtask 10 A (phrase-level polarity prediction) (Rosenthal et al., 2015) and achieved second place (0.6459 ${F_1}^{+/-}$) in Subtask 10 B (message-level

classification) of this competition. Drawing on the ideas of Kalchbrenner et al. (2014), the authors devised a simple convolutional network in which they multiplied pretrained word embeddings with 300 distinct convolutional kernels each of width 5, pooled the maximum value of this multiplication for each of the kernels, and then passed the results of this pooling to a piecewise linear ReLU filter with a densely connected softmax layer. An important aspect of this approach which accounted for a huge part of its success was a special multi-stage training scheme that was used to optimize model's parameters: In the initial stage of this scheme, Severyn and Moschitti first computed Twitter-specific word embeddings by applying the word2vec algorithm to a large Twitter corpus. Afterwards, they pretrained the complete system including the word vectors, convolutional filters, and inter-layer matrices on a big set of noisily labeled microblogs from this collection; and, finally, fine-tuned the parameters of the model on the officially released SemEval data.

Later on, this system was further improved by Deriu et al. (2016), who increased the number of convolutional layers (applying two layers instead of one) and simultaneously trained two such models (using word2vec vectors as input for the first one and passing GloVe embeddings to the second), joining their output at the end and achieving this way 0.671 ${F_1}^{+/-}$ on the SemEval-2015 testset. A similar enhancement was also proposed by Rouvier and Favre (2016), who likewise utilized three different types of embeddings (word2vec, word2vec specific to particular parts of speech, and sentiment-tailored vectors), training separate sets of convolutions for each of these types.

Although convolutional approaches still show competitive scores and are hard to outperform in practice, in recent time, they are gradually being superseded by recurrent neural networks (Xu et al., 2016; Wang et al., 2015b). One of the most prominent such systems has been recently proposed by Baziotis et al. (2017). In their submission to SemEval 2017 (Rosenthal et al., 2017), the authors used two successive bidirectional LSTM units (BiLSTMs). In each of these units, they concatenated the results of the left-to-right recurrence ($\vec{h}_{i_{\rightarrow}}^{(l)} \in \mathbb{R}^{150}$) with the respective outputs of the right-to-left loop ($\vec{h}_{i_{\leftarrow}}^{(l)} \in \mathbb{R}^{150}$) and then passed the result of this concatenation ($\vec{h}_i^{(l)} = [\vec{h}_{i_{\rightarrow}}^{(l)}, \vec{h}_{i_{\leftarrow}}^{(l)}] \in \mathbb{R}^{300}$) to the next layer of the network. After getting the output of the second BiLSTM, they united the states of this unit from all time steps i into a single vector $\vec{a}$ with the help of a special attention mechanism, in which they first multiplied each of the BiLSTM states $\vec{h}_i$ with the respective globally normalized attention score a_i and then took the sum of these weighted vectors over all i

positions:

$$\begin{aligned} \vec{a} &= \sum_{i=1}^{|\mathbf{x}|} a_i \vec{h}_i, \\ \text{where } a_i &= \frac{\exp(e_i)}{\sum_{j=1}^{|\mathbf{x}|} \exp(e_j)}, \\ \text{s.t. } e_i &= tanh\left(\vec{\alpha}\vec{h}_i^{(2)} + \beta_i\right). \end{aligned} \tag{6.5}$$

The $\vec{\alpha}$ and β terms in the above equations denote the attention parameters (score and bias) which are optimized along with other network weights during the training process. To make the final prediction, Baziotis et al. multiplied the attention vector $\vec{a}$ with a matrix W and computed element-wise softmax of this product, getting probability scores for each of the three polarity classes and choosing the label with the maximum score:

$$\hat{y} = \underset{y}{\text{argmax}} \left(softmax(W^\top \vec{a})_y\right). \tag{6.6}$$

With this approach, the authors attained the first first place at SemEval-2014 Task 4 (0.675 $F_1^{+/-}$), being on a par with the system of Cliche (2017) and even outperforming the method of Rouvier (2017) despite the fact that both of these competitors used ensembles of LSTMs and convolutional networks.

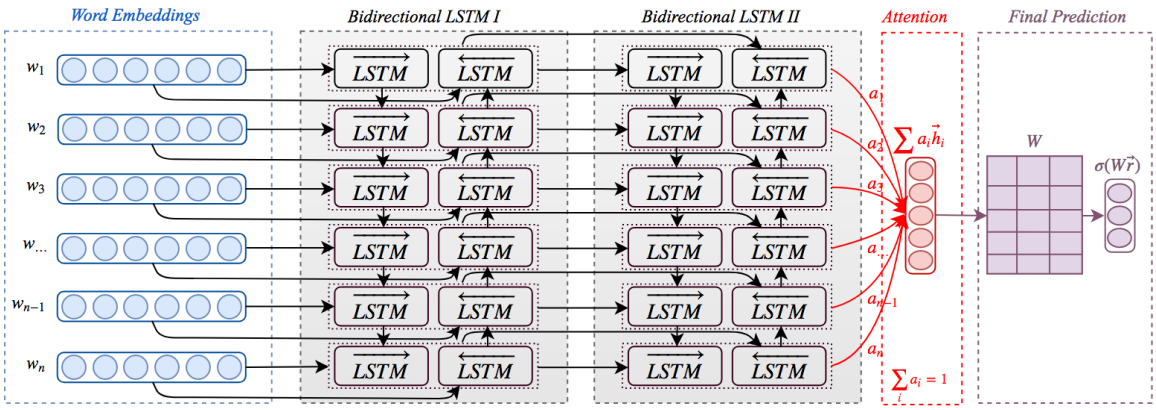


Figure 6.1: Architecture of the neural network proposed by Baziotis et al. (2017)

Even though the approach of Baziotis et al. (2017) represents the current state of the art in sentiment analysis of Twitter and yields extraordinarily good results, in our opinion, this method has yet some potential for improvements. This, first of all, concerns the way how attention coefficients are computed. As we can see from Equation 6.5, the magnitude of the attention score a_i primarily depends on the absolute value of the BiLSTM outputs and the bias term at the i-th position. Albeit this strategy is definitely plausible, assuming the fact that LSTMs shall produce higher scores for opinionated tokens and presupposing that polar terms near the end of the message will usually have a greater influence on the net polarity of

the tweet than opinionated words at its beginning, a crucial prerequisite for this strategy to work is (i) that the LSTM layer can already provide sufficiently reliable results and (ii) the bias terms do not overly boost the importance of irrelevant tokens that just accidentally appeared at favored positions. Unfortunately, both of these prerequisites are rarely fulfilled in practice.

In order to overcome these difficulties, we decided to augment the original architecture of Baziotis et al. (2017) shown in Figure 6.1 with two additional types of attention: *lexicon-* and *context-based* one. In the former type, we estimated the importance weight b_i for position i as the polarity score of the word w_i, obtaining this value from our linear projection lexicon and normalizing it by the sum of polarity scores for all tweet tokens:

$$\begin{aligned}
\vec{b} &= \sum_{i=1}^{|\mathbf{x}|} b_i \vec{h}_i, \\
b_i &= \frac{\exp(f_i)}{\sum_{j=1}^{|\mathbf{x}|} \exp(f_j)}, \\
\text{s.t. } f_i &= \begin{cases} tanh(abs(V[w_i]) + \epsilon) & \text{if } w_i \in V \\ tanh(\epsilon) & \text{otherwise.} \end{cases}
\end{aligned}$$

This way, we hoped to force the network to pay more attention to the BiLSTM outputs that were produced at highly subjective terms rather than favoring arbitrary words in the message.

Another important factor which could notably affect the polarity of a microblog were the so-called *valence shifters* (Polanyi and Zaenen, 2006)—words and phrases such as "kaum" (*hardly*) or "nicht" (*not*) which could significantly change (or even reverse) the semantic orientation of polar terms. To account for these phenomena, we have added another type of attention—the *context-based* one, whose goal was to identify such shifters in the message and give them bigger weights during the attention-based ranking. To discern these elements, we introduced a linear classifier, which had to predict the potential shifting power of a token w_i given its original word embedding $\vec{w}_i$ and the LSTM output of its parent in the dependency tree multiplied with the lexicon-based attention score of that parent ($\vec{b}_p := b_p \vec{h}_p$). To keep the resulting attention scores within an appropriate range, we again used the same *tanh* transformation and global normalization over all positions as we did for the previous two types:

$$\begin{aligned}
\vec{c} &= \sum_{i=1}^{|\mathbf{x}|} c_i \vec{h}_i, \\
c_i &= \frac{\exp(g_i)}{\sum_{j=1}^{|\mathbf{x}|} \exp(g_j)}, \\
g_i &= \tanh\left(C[\vec{w}_i, \vec{b}_p]^\top\right).
\end{aligned}$$

The C term in the above equation represents a context-based attention matrix $\mathbb{R}^{200\times100}$, the $\vec{w}_i$ variable denotes the word embedding of the i-th token, and the $\vec{b}_p$ term stands for the value of the vector $\vec{b}$ (the result of the lexicon-based attention from Equation 6.5) at position p (the index of the syntactic parent of w_i). With this classifier, we hoped to amplify the importance of shifting words in cases when the immediate syntactic ancestors of these tokens were highly opinionated expressions, e.g., "Er hat die Prüfung kaum bestanden" (*He hardly passed the exam*) or "Ich mag den neuen Bundesminister nicht" (*I do not like the new federal minister*), but ignore them if they did not relate to any subjective term.

At last, to make the final prediction, we concatenated the outputs of the three attention layers into a single matrix $A \in \mathbb{R}^{3\times100}$ and multiplied it with a vector $\vec{w} \in \mathbb{R}^{1\times100}$, applying softmax normalization at the end:

$$\begin{aligned}\vec{o} &= softmax\left(A^\top \vec{w}\right), \text{ where}\\ A &= [\vec{a}, \vec{b}, \vec{c}].\end{aligned}$$

Since introducing additional attention types increased the number of model parameters, we removed one of the intermediate Bi-LSTM layers in the network to counterbalance this effect and report our results for both settings: using one and two Bi-LSTM units (denoted as $\text{LBA}^{(1)}$ and $\text{LBA}^{(2)}$ respectively). The final architecture of our approach is shown in Figure 6.2.

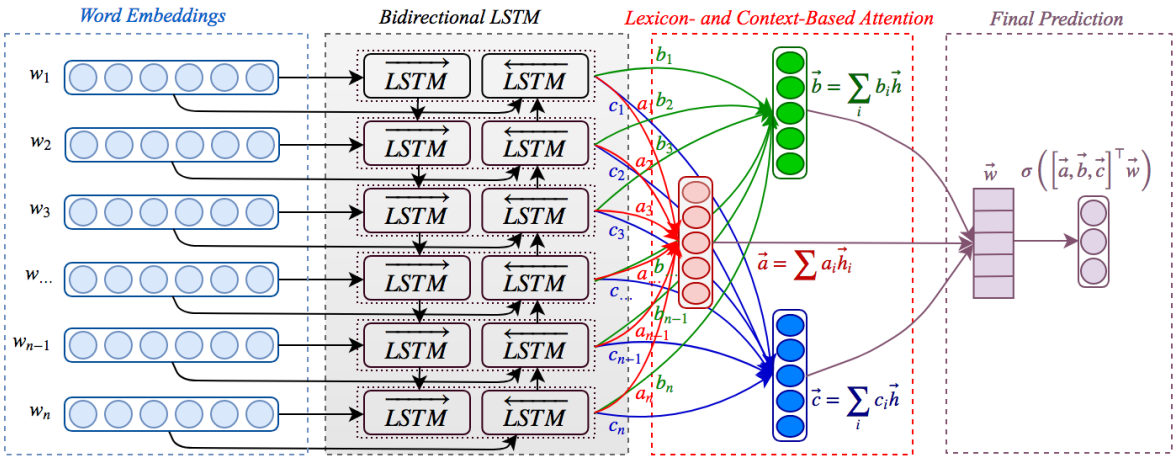


Figure 6.2: Architecture of the neural network with lexicon- and context-based attention

To evaluate the performance of the presented methods and also test our lexicon-based attention mechanism on the PotTS and SB10k datasets, we reimplemented the approaches

of Yessenalina and Cardie (2011), Socher et al. (2011, 2012, 2013), Severyn and Moschitti (2015b), and Baziotis et al. (2017), and applied them along with our own solution to the aforementioned corpora. For the sake of uniformity and simplicity, we used task-specific word embeddings of size $\mathbb{R}^{100}$ in all systems, optimizing these vectors with other network parameters during the training. Moreover, we also unified the final activation parts and cost functions of all networks, using a densely connected softmax layer as the last component of each classifier and optimizing their weights w.r.t. the categorical hinge loss on the training data, picking the values that yielded the highest accuracy on the validation set.

Method	**Positive**			**Negative**			**Neutral**			**Macro**	**Micro**
	Precision	Recall	F_1	Precision	Recall	F_1	Precision	Recall	F_1	F_1	F_1
PotTS											
Y&C	0.45	**1.0**	0.62	0.0	0.0	0.0	0.0	0.0	0.0	0.308	0.446
RAE	0.64	0.78	0.7	0.38	0.04	0.08	0.57	0.68	0.62	0.389	0.605
MVRNN	0.45	**1.0**	0.62	0.0	0.0	0.0	0.0	0.0	0.0	0.308	0.446
RNTN	0.45	0.87	0.59	0.19	0.02	0.03	0.32	0.1	0.15	0.312	0.428
SEV	0.73	0.79	0.76	**0.41**	**0.52**	**0.46**	**0.72**	0.55	0.62	**0.608**	0.651
BAZ	0.45	**1.0**	0.62	0.0	0.0	0.0	0.0	0.0	0.0	0.308	0.446
LBA$^{(1)}$	**0.82**	0.73	**0.77**	0.0	0.0	0.0	0.56	**0.92**	**0.69**	0.387	**0.662**
LBA$^{(2)}$	0.45	**1.0**	0.62	0.0	0.0	0.0	0.0	0.0	0.0	0.308	0.446
SB10k											
Y&C	0.0	0.0	0.0	0.0	0.0	0.0	0.62	**1.0**	0.77	0.0	0.622
RAE	0.63	0.57	0.6	0.0	0.0	0.0	**0.75**	0.94	0.83	0.299	0.721
MVRNN	0.0	0.0	0.0	0.0	0.0	0.0	0.62	**1.0**	0.77	0.0	0.622
RNTN	0.2	0.03	0.05	0.07	0.01	0.02	0.62	0.94	0.75	0.033	0.594
SEV	0.0	0.0	0.0	0.0	0.0	0.0	0.62	**1.0**	0.77	0.0	0.622
BAZ	0.75	0.47	0.58	0.0	0.0	0.0	0.71	0.98	0.83	0.291	0.72
LBA$^{(1)}$	0.72	**0.58**	**0.64**	0.0	0.0	0.0	0.74	0.97	**0.84**	**0.321**	**0.737**
LBA$^{(2)}$	**0.76**	0.49	0.6	0.0	0.0	0.0	0.72	0.98	0.83	0.298	0.723

Table 6.8: Evaluation of DL-based CGSA methods
Y&C – Yessenalina and Cardie (2011), RAE – Recursive Auto-Encoder (Socher et al., 2011), MVRNN – Matrix-Vector RNN (Socher et al., 2012), RNTN – Recursive Neural-Tensor Network (Socher et al., 2013), SEV – Severyn and Moschitti (2015b), BAZ – Baziotis et al. (2017), LBA$^{(1)}$ – lexicon-based attention with one Bi-LSTM layer, LBA$^{(2)}$ – lexicon-based attention with two Bi-LSTM layers

The results of this evaluation are shown in Table 6.8. As we can see from the figures, the LBA method performs fairly well, especially on the positive and neutral classes where it achieves the best F_1-benchmarks on both datasets and also attains the highest overall micro-averaged F_1-scores on all test samples (0.662 on PotTS and 0.737 on SB10k). Even though our approach also yields the best macro-averaged result on the SB10k set (0.321 F_1), it seems to face a major difficulty with the extreme label skewness of this corpus, failing to predict any negative tweet in the test set. This problem, in general, appears to be an insurmountable hurdle for almost all other compared systems, especially the matrix-space, MVRNN, and convolutional approaches, which eventually end up predicting only the most

common neutral label for all messages in this dataset. A single notable exception to this tendency is the recursive neural tensor approach of Socher et al. (2013), which succeeds in classifying some of the negative instances and also predicts positive and neutral labels, but whose precision and recall are still far below an acceptable level.

A similar, though less severe situation is also observed on the PotTS corpus. This time, the Y&C, MVRNN, BAZ, and LBA$^{(2)}$ methods lapse into always predicting only the most frequent positive class. Other systems, however, perform much better, especially the approach of Severyn and Moschitti (2015b), which does an extraordinarily good job at classifying negative messages, reaching remarkable 0.46 F_1 on this subset and also attaining the best macro-average score (0.608) on all tweets due to its competitive performance on positive and neutral microblogs. Nevertheless, even the best-performing DL systems (SEV and LBA) lag far behind the traditional supervised machine-learning method of Mohammad et al. (2013), and barely outperform the lexicon-based approach of Hu and Liu (2004) in terms of the micro-averaged F_1 on SB10k. Two possible explanations for these rather mediocre scores, which we could think of, were a bad starting point of the parameters, which prevented the optimizers from finding the optimal solution to the optimization objective, or an insufficient amount of training data, which lead to an extreme overfitting of the training set, but poor generalization to unseen examples. We should now investigate the former of these factors in detail.

6.5.1 Effect of Word Embeddings

Similarly to the previous chapters, we decided to replace randomly initialized word vectors in the very first layer of the vector-based neural networks with pretrained word2vec embeddings, keeping this parameter fixed during the optimization. As we can see from the figures in Table 6.9, this operation leads to a significant improvement of the results for almost all classifiers except for the recursive auto-encoder and convolutional approach of Severyn and Moschitti (2015b), where it slightly lowers the micro-averaged F_1-score in the former case (from 0.605 to 0.55) and considerably worsens the macro-averaged F_1 (from 0.608 to 0.36 F_1) of the latter system. Nonetheless, even despite these exceptional setbacks, the best observed macro-score increases from 0.608 to 0.64 on the PotTS dataset and almost doubles from 0.321 to 0.53 on the SB10k data. A similar situation is observed with the micro-averaged F_1 which rises from 0.662 to 0.69 on PotTS and also improves from 0.737 to 0.75 on the SB10k corpus. Unfortunately, these improvements usually come at the expense of a lower recall of the majority classes (positive and neutral respectively), but the gains in the overall metrics are generally much higher and, first of all, more important than the losses in these single aspects.

Method	Positive			Negative			Neutral			Macro F_1	Micro F_1
	Precision	Recall	F_1	Precision	Recall	F_1	Precision	Recall	F_1		
PotTS											
RAE	$0.58^{-0.06}$	$0.74^{-0.04}$	$0.65^{-0.05}$	$0.34^{-0.04}$	$0.26^{+0.22}$	$0.29^{+0.21}$	$0.59^{+0.02}$	$0.46^{-0.22}$	$0.52^{-0.1}$	$0.47^{+0.08}$	$0.55^{-0.06}$
RNTN	$0.48^{+0.03}$	$0.77^{-0.1}$	0.59	$0.33^{+0.14}$	$0.03^{+0.01}$	$0.06^{+0.03}$	$0.46^{+0.14}$	$0.33^{+0.23}$	$0.38^{+0.01}$	$0.33^{+0.02}$	$0.47^{+0.04}$
SEV	$0.69^{-0.04}$	$0.74^{-0.05}$	$0.72^{-0.04}$	$0.0^{-0.41}$	$0.0^{-0.52}$	$0.0^{-0.46}$	$0.58^{-0.14}$	$0.84^{+0.29}$	$0.69^{+0.07}$	$0.36^{-0.25}$	$0.64^{-0.01}$
BAZ	$0.85^{+0.4}$	$0.61^{-0.39}$	$0.71^{+0.09}$	$0.57^{+0.57}$	$0.32^{+0.32}$	$0.41^{+0.41}$	$0.55^{+0.55}$	$0.87^{+0.87}$	$0.68^{+0.68}$	$0.56^{+0.25}$	$0.65^{+0.2}$
LBA$^{(1)}$	$0.86^{+0.04}$	$0.6^{-0.13}$	$0.71^{-0.06}$	$0.61^{+0.61}$	$0.46^{+0.46}$	$0.53^{+0.53}$	$0.6^{+0.04}$	$0.89^{-0.03}$	$0.72^{+0.03}$	$0.62^{+0.23}$	$0.68^{+0.02}$
LBA$^{(2)}$	$0.84^{+0.39}$	$0.65^{-0.35}$	$0.73^{+0.11}$	$0.57^{+0.57}$	$0.54^{+0.54}$	$0.55^{+0.55}$	$0.63^{+0.63}$	$0.82^{+0.82}$	$0.72^{+0.72}$	$0.64^{+0.33}$	$0.69^{+0.24}$
SB10k											
RAE	$0.61^{-0.02}$	$0.56^{-0.01}$	$0.58^{-0.02}$	$0.29^{+0.29}$	$0.01^{+0.01}$	$0.02^{+0.02}$	$0.74^{-0.01}$	$0.92^{-0.02}$	$0.82^{-0.01}$	0.3	$0.71^{-0.01}$
RNTN	$0.54^{+0.34}$	$0.02^{-0.01}$	$0.04^{-0.01}$	$0.0^{-0.07}$	$0.0^{-0.01}$	$0.0^{-0.02}$	$0.63^{+0.01}$	$1.0^{+0.06}$	$0.77^{+0.02}$	$0.02^{-0.01}$	$0.62^{+0.03}$
SEV	$0.72^{+0.72}$	$0.5^{+0.5}$	$0.59^{+0.59}$	$0.49^{+0.49}$	$0.27^{+0.27}$	$0.35^{+0.35}$	$0.75^{-0.13}$	$0.92^{-0.08}$	$0.82^{+0.05}$	$0.47^{+0.47}$	$0.73^{+0.11}$
BAZ	$0.78^{+0.03}$	$0.51^{+0.04}$	$0.61^{+0.03}$	$0.49^{+0.49}$	$0.42^{+0.42}$	$0.45^{+0.45}$	$0.78^{+0.07}$	$0.91^{-0.07}$	$0.84^{+0.01}$	$0.53^{+0.24}$	$0.75^{+0.03}$
LBA$^{(1)}$	$0.84^{+0.12}$	$0.42^{-0.16}$	$0.56^{-0.08}$	$0.5^{+0.5}$	$0.28^{+0.28}$	$0.36^{+0.36}$	0.74	$0.96^{-0.01}$	0.84	$0.46^{+0.14}$	$0.73^{+0.01}$
LBA$^{(2)}$	$0.79^{+0.03}$	$0.45^{-0.04}$	$0.57^{-0.03}$	$0.57^{+0.57}$	$0.23^{+0.23}$	$0.33^{+0.33}$	$0.74^{+0.02}$	$0.96^{-0.02}$	$0.84^{+0.01}$	$0.45^{+0.15}$	$0.74^{+0.02}$

Table 6.9: Evaluation of DL-based CGSA methods with pretrained word2vec vectors RAE – Recursive Auto-Encoder (Socher et al., 2011), RNTN – Recursive Neural-Tensor Network (Socher et al., 2013), SEV – Severyn and Moschitti (2015b), BAZ – Baziotis et al. (2017), LBA$^{(1)}$ – lexicon-based attention with one Bi-LSTM layer, LBA$^{(2)}$ – lexicon-based attention with two Bi-LSTM layers

In order to see whether these changes would be different if we optimized word representations as well, we reran our experiments once again, initializing word vectors with word2vec embeddings as before, but allowing them to be updated during the training. Moreover, to approximate task-specific representations of words which were missing from the training set, we also computed the optimal transformation matrix for converting the original word2vec vectors into optimized sentiment embeddings using the method of the ordinary least squares, and used this matrix to derive task-specific vectors during the testing.

As suggested by the results in Table 6.10, these modifications improve the results even further, setting a new record of the macro-averaged F_1-scores on the PotTS corpus (0.69 F_1), and pushing our LBA$^{(1)}$ system even above its most challenging competitors. A similar effect is also observed with other systems, first of all BAZ and LBA$^{(2)}$, which yield similarly good results for all polarities. Nevertheless, akin to the previous case, these improvements usually cause a drop of the recall for the most frequent class of the respective dataset, which is especially severe for the system of Baziotis et al. on the PotTS data (-0.28 on positive messages) and the LBA$^{(1)}$ approach on the SB10k test set (-0.17 on neutral tweets). Furthermore, the convolutional system of Severyn and Moschitti and recursive neural tensor approach of Socher et al. (2013) fail to predict any negative tweet on PotTS and SB10k respectively, which also leads to a notable drop of their overall macro-F_1–values. These drops, however, are rather exceptional, as the same system of Severyn and Moschitti shows an extraordinary big boost of the results on the SB10k corpus (+0.45 macro-F_1 and +0.1

micro-F_1–score), and the macro-averaged F_1-values of all recurrent methods also become twice as high as in the case of randomly initialized word vectors.

Method	Positive			Negative			Neutral			Macro F_1	Micro F_1
	Precision	Recall	F_1	Precision	Recall	F_1	Precision	Recall	F_1		
PotTS											
RAE	$0.61^{-0.03}$	$0.61^{-0.17}$	$0.61^{-0.09}$	$0.22^{-0.16}$	$0.01^{-0.03}$	$0.03^{-0.05}$	$0.48^{-0.09}$	$0.72^{-0.04}$	$0.57^{-0.05}$	$0.32^{-0.07}$	$0.54^{-0.07}$
RNTN	0.45	$0.82^{-0.05}$	0.59	$0.24^{+0.05}$	$0.06^{-0.04}$	$0.1^{-0.07}$	$0.43^{+0.09}$	$0.17^{+0.07}$	$0.24^{+0.09}$	$0.34^{+0.03}$	$0.44^{-0.01}$
SEV	0.73	$0.74^{-0.05}$	$0.74^{-0.02}$	$0.0^{-0.41}$	$0.0^{-0.52}$	$0.0^{-0.46}$	$0.56^{-0.16}$	$0.84^{+0.29}$	$0.68^{+0.06}$	$0.37^{-0.24}$	$0.64^{-0.01}$
BAZ	$0.82^{+0.37}$	$0.72^{-0.28}$	$0.77^{+0.15}$	$0.62^{+0.62}$	$0.49^{+0.49}$	$0.55^{+0.55}$	$0.68^{+0.68}$	$0.85^{+0.85}$	$0.76^{+0.76}$	$0.66^{+0.35}$	$0.73^{+0.28}$
LBA$^{(1)}$	$0.76^{-0.06}$	$0.84^{+0.11}$	$0.79^{+0.02}$	$0.6^{+0.6}$	$0.56^{+0.56}$	$0.58^{+0.58}$	$0.75^{+0.19}$	$0.68^{-0.24}$	$0.72^{+0.03}$	$0.69^{+0.3}$	$0.73^{+0.07}$
LBA$^{(2)}$	$0.84^{+0.39}$	$0.73^{-0.27}$	$0.78^{+0.16}$	$0.57^{+0.57}$	$0.48^{+0.48}$	$0.53^{+0.53}$	$0.66^{+0.66}$	$0.82^{+0.82}$	$0.73^{+0.73}$	$0.65^{+0.34}$	$0.72^{+0.27}$
SB10k											
RAE	$0.5^{-0.13}$	$0.73^{+0.16}$	$0.59^{-0.01}$	$0.35^{+0.35}$	$0.06^{+0.06}$	$0.1^{+0.1}$	$0.8^{+0.05}$	$0.8^{-0.14}$	$0.8^{-0.03}$	$0.35^{+0.15}$	$0.68^{-0.04}$
RNTN	$0.0^{-0.02}$	$0.0^{-0.03}$	$0.0^{-0.05}$	$0.0^{-0.07}$	$0.0^{-0.01}$	$0.0^{-0.02}$	0.62	$1.0^{-0.06}$	$0.77^{-0.02}$	$0.0^{-0.03}$	$0.62^{+0.03}$
SEV	$0.64^{+0.64}$	$0.58^{+0.58}$	$0.61^{+0.61}$	$0.51^{+0.51}$	$0.21^{+0.21}$	$0.3^{+0.3}$	$0.76^{+0.14}$	$0.89^{-0.11}$	$0.82^{+0.05}$	$0.45^{+0.45}$	$0.72^{+0.1}$
BAZ	$0.72^{+0.03}$	$0.59^{+0.12}$	$0.65^{+0.07}$	$0.53^{+0.53}$	$0.33^{+0.33}$	$0.41^{+0.41}$	$0.79^{+0.08}$	$0.91^{-0.07}$	$0.84^{+0.01}$	$0.53^{+0.24}$	$0.75^{+0.03}$
LBA$^{(1)}$	$0.6^{-0.12}$	$0.72^{+0.14}$	$0.66^{+0.02}$	$0.47^{+0.47}$	$0.42^{+0.42}$	$0.44^{+0.44}$	$0.84^{+0.1}$	$0.8^{-0.17}$	$0.82^{+0.02}$	$0.55^{+0.23}$	$0.73^{-0.01}$
LBA$^{(2)}$	$0.72^{-0.04}$	$0.57^{+0.08}$	$0.64^{+0.04}$	$0.55^{+0.55}$	$0.39^{+0.39}$	$0.46^{+0.46}$	$0.79^{+0.07}$	$0.9^{-0.08}$	$0.84^{+0.01}$	$0.55^{+0.25}$	$0.75^{+0.03}$

Table 6.10: Evaluation of DL-based CGSA methods with pretrained embeddings and the least-squares fallback

RAE – Recursive Auto-Encoder (Socher et al., 2011), RNTN – Recursive Neural-Tensor Network (Socher et al., 2013), SEV – Severyn and Moschitti (2015b), BAZ – Baziotis et al. (2017), , LBA$^{(1)}$ – lexicon-based attention with one Bi-LSTM layer, LBA$^{(2)}$ – lexicon-based attention with two Bi-LSTM layers

6.5.2 Error Analysis

Before we proceed with the evaluation of the other factor (a larger size of the training set), let us first analyze some errors that were specific to each of the classifiers trained with task-specific embeddings and the least-squares fallback.

Since interpreting and understanding the results of deep learning systems is a generally complex task due to a big number of model parameters and unobvious correlations between them, we decided to use the LIME package (Ribeiro et al., 2016)—a recently proposed model-agnostic interpretation tool—to get a better intuition about the reasons of the classifiers' decisions. To derive an explanation for a particular prediction, LIME randomly removes or perturbs parts of the input (in our case, tokens), estimating which of these modifications lead to the biggest changes in the output probabilities, and assigns corresponding class-specific association scores to each of the changed parts. The higher this score, the more predictive is the given feature of that particular label. For the sake of vividness, we have highlighted all tokens that, according to LIME, were associated with the neutral class as white, marked

negative attributes with the blue background, and highlighted positively connoted words in green, reflecting the respective association strength with a higher color brightness.

The first incorrect prediction shown in Example 6.5.1 was made by the RAE system of Socher et al. (2011). As we can see from the visualization, the model has correctly recognized the positive term "gefällt" (*to like*), but, unfortunately, this word is the only one which contributes to the right decision, and its learned weight is obviously not enough to outdo the effect of multiple neutral and negative items, such as "Grün" (*green*), "Schwarz" (*black*), and most surprisingly "PosSmiley%" (*PosSmiley%*), which unexpectedly is stronger associated with the negative semantic orientation than with the positive class. As a consequence of this, the classifier erroneously predicts the NEUTRAL label for the whole message, falling against the prevalence of allegedly objective terms.

Example 6.5.1 (Error Made by the RAE System)

Tweet: Grün - Schwarz in meinem Bundesland. Gefällt mir doch sehr %PosSmiley

Green - Black in my state. Yet , I like it so much %PosSmiley

Gold Label: **positive**

Predicted Label: **neutral***

A similar situation is also observed with the recurrent neural tensor, whose sample error is shown in Example 6.5.2. As we can see from the analysis, the bias towards the neutral class is even more pronounced this time as virtually all of the terms in the tweet are highlighted in white. The only word which shows a minimal negative connotation is "tumblr", which indeed appeared twice in a negative tweet, two times in neutral messages, and once in a positive microblog in the training corpus. Nonetheless, even for this term the skewness towards the neutral orientation is still ten times bigger than its association with the negative polarity ($1.4e^{-4}$ versus $1.5e^{-5}$), which can be explained by the general prevalence of neutral messages in the SB10k data.

Example 6.5.2 (Error Made by the RNTN System)

Tweet: tumblr people sind meine lieblings people %PosSmiley

tumblr people are my favorite people %PosSmiley

Gold Label: **positive**

Predicted Label: **neutral***

A slightly different behavior is shown by the method of Severyn and Moschitti (2015b) on the PotTS dataset. This time, we can see at least two clearly positive words ("ist" [*is*] and "Freund" [*friend*]). However, the former of these terms is an auxiliary copular verb, which can hardly express any polarity, since it usually plays a purely grammatical role and

lacks any distinct lexical meaning. Nevertheless, the latter word ("Freund" [*friend*]) indeed conveys a positive feeling of its prepositional argument ("Iran") towards the subject of the sentence ("Syrien" [*Syria*]), but this positive effect is nullified by the author's statement that this friendship poses a problem. Unfortunately, the word "Problem" (*problem*) is recognized only as a neutral marker, just like many other terms in this microblog.

Example 6.5.3 (Error Made by the SEV System)
Tweet: Syrien ist Freund von Iran , das ist das Problem ! annewill
Syria is a friend of Iran . That 's the problem ! annewill

Gold Label: **negative**
Predicted Label: **neutral***

In Example 6.5.4, we can see another error made by the system of Baziotis et al. (2017). This time, again, we observe the prevalence of positive and neutral items, with the only exception being the term "meinen" (*my*), which, according to the classifier, indicates the negative polarity even though it has a clear objective possessive sense. Apart from this term, we also can notice several inaccuracies at recognizing positive and neutral features: For example, the pronominal adverb "darin" (*in it*) is the strongest positive trait, whose predictiveness is even higher than the scores of the words "singen" (*to sing*) and "Liebeslied" (*love song*). This contradicts the fact that pronominal adverbs by themselves do not express any semantic orientation, all the more as in this case the antecedent of the adverb—the noun "Kleiderschrank" (*wardrobe*)—has been already recognized as a neutral item. On the other hand, the modal verb "wollte" (*wanted*) is considered as an objective term, although it has a slight positive connotation as it expresses a wish of the author.

Example 6.5.4 (Error Made by the BAZ System)
Tweet: Wollte meinen Kleiderschrank aufräumen ... sitze nun darin und singe Liebeslieder ...
Wanted to clean up my wardrobe ... Now sitting in it and singing love songs ...

Gold Label: **neutral**
Predicted Label: **positive***

Finally, Example 6.5.5 shows an incorrect prediction of our lexicon-based attention approach. In contrast to the previous two methods, the positive information is much more condensed in this case and represented by a single term "super". Surprisingly, this term outweighs a whole bunch of neutral features such as "gerade" (*right now*), "Lust haben" (*to be up to*), "was" (*something*) etc. Admittedly, the first part of this message indeed expresses

a positive attitude of the author, but this effect is invalidated by the second clause which shows the impossibility of that wish.

Example 6.5.5 (Error Made by the LBA System)
Tweet: Gerade super Lust , mit Carls Haaren was zu machen aber ca 300 km Distanz halten mich davon ab .
Right now super up to do something with Carl 's hair, but ca. 300 km distance keep me off from this.

Gold Label: **neutral**
Predicted Label: **positive***

6.6 Evaluation

Now that we have familiarized ourselves with the peculiarities and results of the most prominent sentiment analysis approaches from all method groups (lexicon-, machine-learning– and deep-learning–based ones), let us have a closer look at how changing different common parameters of these methods might affect their performance. In particular, we would like to see whether increasing the amount of training data, switching to a different type of sentiment lexicon, or using unnormalized text as input could improve or, vice versa, lower the classification scores.

6.6.1 Distant Supervision

The first avenue that we are going to explore in this evaluation is the effect of distantly supervised data—an additional collection of training tweets which have been automatically labeled with sentiment tags based on the occurrence of some sufficiently reliable formal criterions, such as emoticons or polar hashtags.

Among the first who proposed the idea of training a sentiment classifier on a larger corpus of automatically annotated messages was Read (2005) who gathered a set of 766,000 Usenet posts containing frownies or smileys, assigned a polarity label to each of these posts, judging by the type of the emoticons, and subsequently used a subset of these documents (22,000 posts) to optimize a Naïve Bayes and SVM system. Even though these classifiers could achieve a considerable accuracy (up to 70%) on predicting the noisy labels of the remaining posts, they could not generalize to texts from other genres (movie reviews and newswire articles) where they hardly outperformed the random-chance baseline.

With the onset of the Twitter era, this idea of distant supervision has experienced its renaissance, with one of the first revival attempts made by Go et al. (2009) who collected

a corpus of 800,000 positive and 800,000 negative microblogs, also considering emoticons as their noisy labels. After stripping these smileys off from the text, the authors trained three independent ML-classifiers (Naïve Bayes, Maximum Entropy, and Support Vector Machines) on these data, achieving their best results (82.7% accuracy) with the NB and MaxEnt systems which utilized unigrams and bigrams as features.

In a similar way, Pak and Paroubek (2010) scraped a collection of 300,000 noisily labeled tweets, ensuring an even distribution of positive, negative, and neutral messages, and trained a Naïve Bayes classifier, using reliable part-of-speech and n-gram features.[9] With this approach, the authors attained an accuracy of slightly more than 60% on the manually annotated test set of the Go et al.'s corpus, but demonstrated a particular utility of bigrams, negation rules, and feature pruning heuristics.

Other notable works on opinion mining with distant supervision were introduced by Barbosa and Feng (2010) who harnessed three publicly available sentiment websites (Twendz, Twitter Sentiment, and TweetFeel) to bootstrap the labels for their downloaded messages; Davidov et al. (2010) who tried to predict the hashtags and emoticons occurring in tweets using a k-NN classifier; and Kouloumpis et al. (2011) who optimized an AdaBoost system (Schapire and Singer, 2000) on two large collections of noisily labeled microblogs—the emoticon tweebank of Go et al. (2009) and the Edinburgh hashtag corpus,[10] achieving a macro-averaged F_1-score of 0.68 on this three-class prediction task.

In order to check the effect of such noisily annotated data on our tested methods, we also automatically labeled all messages from the German Twitter Snapshot (Scheffler, 2014) using the occurrences of smileys as clues: In particular, we considered a microblog as POSITIVE if its normalized version contained the token `%PositiveSmiley` with no other facial expressions. Similarly to this, we considered a message as NEGATIVE if the only emoticon which appeared in this tweet was `%NegativeSmiley`. Furthermore, we classified all posts which contained both types of smileys as MIXED, skipping them during further processing, and assigned the rest of the messages to the NEUTRAL polarity.

A detailed breakdown of the final distribution is given in Table 6.1 at the beginning of this chapter. As we could see from the table, objective messages clearly dominate the whole

[9] Pak and Paroubek (2010) determined the reliability of a feature f using a special *salience* metric, which was defined as a negative ratio between the minimum and maximum conditional probabilities of this feature belonging to different target classes:

$$salience(f) = \frac{1}{N} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} 1 - \frac{\min(P(f, s_i), P(f, s_j))}{\max(P(f, s_i), P(f, s_j))},$$

where the N term denotes the number of training examples, and s_i means the sentiment class of the i-th training instance.

[10] `http://demeter.inf.ed.ac.uk`

dataset, accounting for 84% of the data; followed by the POSITIVE class, which makes up 14% of the tweets. Similarly to the manually annotated PotTS and SB10k corpora, the NEGATIVE semantic orientation represents the absolute minority in this collection, featuring only ≈ 350K microblogs (0.76%).

Since it was impossible to utilize the whole snapshot for the training due to limited computational resources (only reading the dataset into memory would require 9.3Gb RAM, not to mention the space required for storing the embeddings and features), we confined ourselves to one sixth of these data, which resulted in 4M messages. Moreover, to mitigate the extreme skewness of this corpus, we downsampled positive and neutral tweets to get an equal number of instances for all classes (59K microblogs for each polarity) and used these examples in addition to the manually analyzed PotTS and SB10k tweets.

Since lexicon-based approaches were mostly independent of the training set, we decided to rerun our experiments only with ML- and fastest DL–based methods (RNN, SEV, BAZ, and LBA),[11] which still incurred running times up to five days for some systems. The results of this evaluation are shown in Table 6.11.

As we can see from the scores, apart from the improved precision of positive tweets and higher recall of negative microblogs, adding noisily labeled messages to the training set has a strong negative effect on the results of all systems, with the biggest drops demonstrated by the approach of Baziotis et al. (-0.5 macro-F_1 [-0.43 micro-F_1] on the PotTS corpus and -0.34 macro-F_1 [-0.51 micro-F_1] on the SB10k dataset) and our own LBA$^{(2)}$ solution (-0.42 macro-F_1-score [-0.5 micro-F_1] on the PotTS test set and -0.43 macro-F_1 [-0.61 micro-F_1] on the SB10k data), which both fail to predict any neutral message on PotTS and always assign the same polarity to all SB10k tweets. Less severe, but still substantial degradation is also observed with the machine-learning systems of Mohammad et al. and Günther and Furrer as well as our DL-based LBA$^{(1)}$ method, whose macro-averaged F_1-values go down by 0.15, 0.12, and 0.26 points on the former corpus and sink by 0.21, 0.15, 0.21 respectively on the latter dataset. The micro-averaged F_1-results of these methods, however, decrease to a much smaller degree, since the main drops happen on the negative class, which is by far the least represented polarity in both corpora. The micro-averages of the remaining systems seem to be affected even less, but are still worse than the results obtained without the snapshot tweets. We hypothesize that the main reason for this decrease is a substantial difference between the class distributions in noisily annotated training data and manually labeled test sets, which overly bias the classifiers' predictions.

[11] In all subsequent evaluation experiments with DL-based systems, we will use pre-trained word2vec vectors (if applicable) with the least-squares fallback, and compare the results of these approaches to the respective scores in Table 6.10.

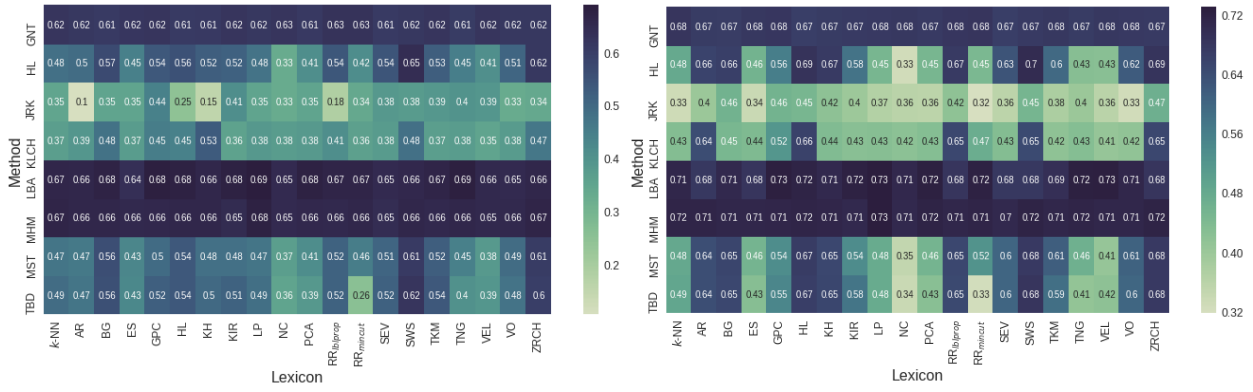
Method	Positive			Negative			Neutral			Macro F_1	Micro F_1
	Precision	Recall	F_1	Precision	Recall	F_1	Precision	Recall	F_1		
PotTS											
GMN	$0.8^{+0.13}$	$0.34^{-0.39}$	$0.48^{-0.22}$	$0.2^{-0.15}$	$0.29^{+0.14}$	$0.24^{-0.03}$	$0.53^{-0.07}$	$0.79^{+0.07}$	$0.63^{-0.03}$	$0.36^{-0.01}$	$0.49^{-0.12}$
MHM	$0.86^{+0.07}$	$0.59^{-0.18}$	$0.7^{-0.08}$	$0.31^{-0.27}$	$0.39^{-0.17}$	$0.35^{-0.22}$	$0.55^{-0.18}$	$0.68^{-0.08}$	$0.61^{-0.13}$	$0.52^{-0.15}$	$0.59^{-0.14}$
GNT	$0.86^{+0.15}$	$0.6^{-0.2}$	$0.71^{-0.04}$	$0.26^{-0.29}$	$0.31^{-0.14}$	$0.28^{-0.22}$	$0.53^{-0.15}$	$0.68^{-0.05}$	$0.59^{-0.06}$	$0.5^{-0.12}$	$0.57^{-0.1}$
RAE	$0.68^{+0.07}$	$0.31^{-0.3}$	$0.43^{-0.18}$	$0.25^{+0.03}$	$0.46^{+0.45}$	$0.32^{+0.29}$	$0.49^{+0.01}$	$0.61^{-0.11}$	$0.54^{-0.03}$	$0.38^{+0.06}$	$0.45^{-0.09}$
SEV	$0.87^{+0.14}$	$0.51^{-0.23}$	$0.64^{-0.1}$	$0.27^{+0.27}$	$0.49^{+0.49}$	$0.35^{+0.35}$	$0.55^{-0.01}$	$0.58^{-0.26}$	$0.56^{-0.12}$	$0.49^{+0.12}$	$0.53^{-0.11}$
BAZ	$0.0^{-0.82}$	$0.0^{-0.72}$	$0.0^{-0.77}$	$0.19^{-0.43}$	$1.0^{+0.51}$	$0.32^{-0.23}$	$0.0^{-0.68}$	$0.0^{-0.85}$	$0.0^{-0.76}$	$0.16^{-0.5}$	$0.19^{-0.43}$
LBA$^{(1)}$	$0.48^{-0.28}$	$0.88^{+0.04}$	$0.62^{-0.17}$	$0.25^{-0.35}$	$0.23^{-0.33}$	$0.24^{-0.34}$	$0.0^{-0.75}$	$0.0^{-0.68}$	$0.0^{-0.72}$	$0.43^{-0.26}$	$0.44^{-0.29}$
LBA$^{(2)}$	$0.91^{+0.07}$	$0.08^{-0.65}$	$0.14^{-0.64}$	$0.19^{-0.38}$	$0.99^{+0.51}$	$0.32^{-0.21}$	$0.0^{-0.66}$	$0.0^{-0.82}$	$0.0^{-0.73}$	$0.23^{-0.42}$	$0.22^{-0.5}$
SB10k											
GMN	$0.71^{+0.06}$	$0.27^{-0.18}$	$0.4^{-0.13}$	$0.24^{-0.14}$	$0.11^{+0.03}$	$0.15^{+0.02}$	$0.71^{-0.01}$	$0.96^{+0.03}$	$0.82^{-0.01}$	$0.27^{-0.06}$	$0.68^{-0.02}$
MHM	$0.77^{+0.06}$	$0.4^{-0.25}$	$0.53^{-0.15}$	$0.61^{-0.1}$	$0.1^{-0.3}$	$0.18^{-0.27}$	$0.71^{-0.09}$	$0.97^{-0.1}$	$0.82^{-0.02}$	$0.35^{-0.21}$	$0.71^{-0.04}$
GNT	$0.77^{+0.1}$	$0.39^{-0.23}$	$0.52^{-0.12}$	$0.25^{-0.19}$	$0.13^{-0.15}$	$0.17^{-0.17}$	$0.71^{-0.07}$	$0.92^{+0.05}$	$0.8^{-0.02}$	$0.34^{-0.15}$	$0.68^{-0.04}$
RAE	$0.44^{-0.06}$	$0.27^{-0.51}$	$0.34^{-0.25}$	$0.24^{-0.11}$	$0.59^{+0.53}$	$0.34^{+0.24}$	$0.78^{-0.02}$	$0.62^{-0.18}$	$0.69^{-0.11}$	$0.34^{-0.01}$	$0.54^{-0.14}$
SEV	0.64	$0.39^{-0.19}$	$0.49^{-0.12}$	$0.34^{-0.17}$	$0.12^{-0.09}$	$0.18^{-0.12}$	$0.7^{-0.06}$	$0.9^{+0.01}$	$0.78^{-0.04}$	$0.33^{-0.12}$	$0.69^{-0.03}$
BAZ	$0.24^{-0.48}$	$1.0^{+0.41}$	$0.38^{-0.27}$	$0.0^{-0.53}$	$0.0^{-0.33}$	$0.0^{-0.41}$	$0.0^{-0.79}$	$0.0^{-0.91}$	$0.0^{-0.84}$	$0.19^{-0.34}$	$0.24^{-0.51}$
LBA$^{(1)}$	$0.64^{+0.04}$	$0.43^{-0.29}$	$0.52^{-0.14}$	$0.59^{+0.12}$	$0.09^{-0.33}$	$0.16^{-0.28}$	$0.71^{-0.13}$	$0.93^{+0.13}$	$0.8^{-0.02}$	$0.34^{-0.21}$	$0.69^{-0.04}$
LBA$^{(2)}$	$0.0^{-0.72}$	$0.0^{-0.57}$	$0.0^{-0.64}$	$0.14^{-0.41}$	$1.0^{+0.61}$	$0.25^{-0.21}$	$0.0^{-0.79}$	$0.0^{-0.9}$	$0.0^{-0.84}$	$0.12^{-0.43}$	$0.14^{-0.61}$

Table 6.11: Results of CGSA methods with distantly supervised data GMN – Gamon (2004), MHM – Mohammad et al. (2013), GNT – Günther et al. (2014), RAE – Recursive Auto-Encoder (Socher et al., 2011), SEV – Severyn and Moschitti (2015b), BAZ – Baziotis et al. (2017), LBA$^{(1)}$ – lexicon-based attention with one Bi-LSTM layer, LBA$^{(2)}$ – lexicon-based attention with two Bi-LSTM layers

6.6.2 Lexicons

Another factor which could significantly affect the results of some systems was the sentiment lexicon that these systems used either directly for computing the polarity of a message (e.g., lexicon-based approaches) or indirectly for extracting some features or assigning attention scores to some parts of the input (e.g., ML- and DL-based techniques). To estimate the effect of this resource, we successively replaced the lexicons which we utilized in our previous experiments with other polarity lists presented in Chapter 4, and recomputed the scores of the tested systems.

As we can see in Figure 6.3, the system of Mohammad et al. (2013) and our own lexicon-based attention approach clearly outperform all other competitors on the PotTS corpus independent of the lexicon they use. The only method which comes at least close to their results—the ML-based classifier of Günther et al. (2014)—is still almost 5% below the average macro-F_1 of these two classifiers . The same also applies to the micro-F_1-scores where the solution of Günther et al. (2014) loses ≈3% on average to the two top performers. Regarding the differences between the MHM and LBA themselves, we can observe a rather mixed relation: The approach of Mohammad et al. (2013) yields better macro-averaged F_1-results



(a) `Macro`-F_1 *(b)* `Micro`-F_1

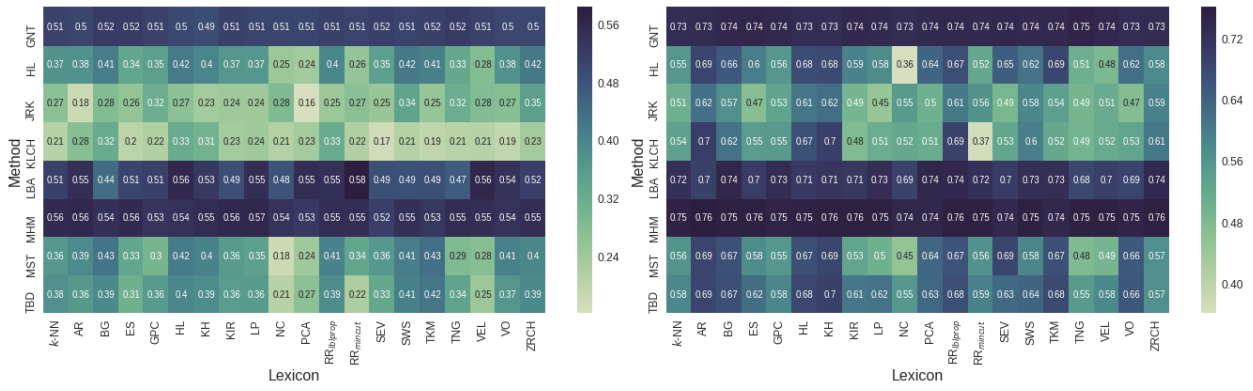
Figure 6.3: Results of CGSA methods with different lexicons on the PotTS corpus

with the lexicons of Esuli and Sebastiani (2005), Vo and Zhang (2016), and Clematide and Klenner (2010), but falls against LBA when used with the polarity lists of Blair-Goldensohn et al. (2008), Waltinger (2010), Hu and Liu (2004), Kiritchenko et al. (2014), Rao and Ravichandran (2009), Takamura et al. (2005), Tang et al. (2014b), and Velikovich et al. (2010) as well as the NWE-based LinProj and PCA lexicons. Moreover, when trained with the polarity list of Tang et al. and our LinProj lexicon, the LBA system also produces the best overall macro-F_1 on this corpus.

These results, however, look slightly differently when we consider the micro-averaged scores. This time, the system of Mohammad et al. outperforms our solution in 8 out of 20 cases, but performs worse than LBA with 4 other polarity lists (GPC, KIR, $\text{RR}_{\text{mincut}}$, and VEL). Nevertheless, our approach still reaches the best overall observed score (0.73) with three tested resources (GPC, LinProj, and VEL).

Regarding the performance of single lexicons, we can see that the best results are achieved with the manually curated SentiWS and Zurich Polarity lists (Remus et al., 2010; Clematide and Klenner, 2010) followed by the dictionary-based approaches of Blair-Goldensohn et al. (2008) and Rao and Ravichandran (2009). The method of the nearest centroids vice versa appears to be of the lowest utility for almost all systems, even though it demonstrated quite acceptable scores in our initial intrinsic evaluation.

A similar situation also holds for the SB10k corpus where the ML-based approaches of Mohammad et al. (2013) and Günther et al. (2014) and our proposed LBA system outperform all other methods in terms of both macro- and micro-averaged F_1-scores. This time, however, the average difference between the macro-results of LBA and GNT is much smaller and amounts to only 0.02% in favor of LBA, which again achieves the best overall macro-F_1 (0.58) in combination with the min-cut lexicon of Rao and Ravichandran (2009). However, our system clearly falls against the latter classifier with respect to the micro-averaged scores, performing worse than it in 16 out of 20 experiments.



(a) $\mathtt{Macro}$-F_1 *(b)* $\mathtt{Micro}$-F_1

Figure 6.4: Results of CGSA methods with different lexicons on the SB10k corpus

The effect of single lexicons is also less pronounced than in the PotTS case as all of the tested polarity lists show a more or less similar behavior, especially regarding the macro-averaged F_1-score. In terms of the micro-F_1, however, we can observe that dictionary-based lists—especially those of Awadallah and Radev (2010), Blair-Goldensohn et al. (2008), Hu and Liu (2004), and Kim and Hovy (2004)—lead to altogether better scores than corpus- and NWE-based resources.

6.6.3 Text Normalization

Finally, the last aspect that we are going to analyze in this evaluation is the effect of the text normalization which we applied to the input messages before passing them to the classifiers. To verify the utility of this step, we rerun all experiments from the initial sections, using the original Twitter messages instead of their preprocessed forms, and recalculated the results of the tested systems.

As we can see from the figures in Table 6.12, switching off the normalization has a strong negative effect on the scores of almost all approaches except for the methods of Yessenalina et al. (2010) and Socher et al. (2012, 2013), which notoriously keep predicting the majority class in most of the cases in the same way as they did it before. Apart from this, we can notice that the lexicon-based systems (HL, JRK, KLCH, MST, and TBD) suffer the greatest loss in terms of both macro- and micro-averaged F_1-scores on the PotTS corpus (up to -0.25 macro- and and -0.22 micro-F_1). A closer look at their errors revealed that this deterioration is mostly due to the increased variety of different emoticons in the dataset (which were typically unified during the preprocessing) and the absence of these forms in the utilized polarity list. The second biggest quality drop is demonstrated by the DL-based approaches BAZ, LBA$^{(1)}$, and LBA$^{(2)}$, which apparently also got confused by the higher lexical variety of the input and failed to optimize all their internal parameters to properly

fit this diversity. The remaining DL- and ML-based classifiers (especially those of MHM, GNT, and RNTN) seem to be more resistant to the introduced changes, but still show a decrease by up to 0.04 macro- and 0.08 micro-F_1. The only exception in this case is the MVRNN system of Socher et al. (2012), which slightly improves on the negative and neutral classes, leaving the majority class pitfall. However, this increase appears to be too small to positively influence the overall statistics of this method.

Regarding the breakdown of single polarity classes, we can see that most of the rare improvements affect the recall of positive and neutral messages, with the biggest gains demonstrated by the RAE and RNTN approaches (+0.37 and +0.11 respectively). Other positive changes are fairly sporadic and produced by only few classifiers (first of all MVRNN). Nevertheless, even in these exceptional cases, the improvements are typically so small that they hardly outweigh the decreased scores on other aspects and have virtually no effect on the net results for all classes.

A similar situation also happens on the SB10k corpus, where we can see even fewer improvements (in 10 out of 176 cases). The biggest increase this time (+0.16 recall) is demonstrated by the approach of Baziotis et al. (2017) on the negative class. The remaining growths, however, are much smaller and typically range between one and seven percent. On the other hand, three of the tested methods (Y&C, MVRNN, and RNTN) have not changed their results at all and show exactly the same numbers as they did previously with normalized messages, although, most of the time, these classifiers only predict the majority label anyway. As to the rest of the systems, we can see that their scores are notably lower than in our initial experiments, but the decrease is much smaller in comparison with the PotTS corpus. A sad exception in this case is a major drop of the recall of neutral messages (-0.79) demonstrated by our LBA$^{(1)}$ system, which, in turn, results in a significant decrease of its macro- and micro-averaged F_1-scores (-0.14 and -0.46 respectively). Other approaches (including the sibling method LBA$^{(2)}$) behave much more stable in this regard and their average decrease amounts to -0.06 macro- and -0.03 micro-F_1.

Similarly to the results on the PotTS data, most of the gains are concentrated at the recall of the neutral class (four out of ten improvements), with the other positive changes being rather sporadic and affecting only a few classifiers. Nevertheless, unlike in the previous case, this time, we can even observe a slight improvement of the macro-averaged F_1-measure for one of the systems (the lexicon-based approach of Jurek et al.), but its micro-averaged metric remains mainly unaffected by this increase. However, overall, the vast majority of macro- and micro-F_1-scores show an obvious decline on both datasets, which once again proves the advantage of preprocessing colloquial texts before applying a standard NLP pipeline.

Method	Positive			Negative			Neutral			Macro	Micro
	Precision	Recall	F_1	Precision	Recall	F_1	Precision	Recall	F_1	$F_1^{+/-}$	F_1
PotTS											
HL	$0.63^{-0.12}$	$0.3^{-0.46}$	$0.4^{-0.36}$	$0.46^{-0.07}$	$0.29^{-0.14}$	$0.36^{-0.11}$	$0.41^{-0.26}$	$0.77^{+0.04}$	$0.54^{-0.15}$	$0.38^{-0.24}$	$0.464^{-0.22}$
TBD	$0.65^{-0.12}$	$0.24^{-0.47}$	$0.36^{-0.38}$	$0.46^{-0.08}$	$0.27^{-0.12}$	$0.34^{-0.11}$	$0.41^{-0.22}$	$0.83^{+0.06}$	$0.55^{-0.14}$	$0.348^{-0.25}$	$0.457^{-0.22}$
MST	$0.63^{-0.12}$	$0.29^{-0.43}$	$0.4^{-0.34}$	$0.47^{-0.01}$	$0.34^{-0.13}$	$0.39^{-0.09}$	$0.42^{-0.26}$	$0.77^{+0.05}$	$0.54^{-0.16}$	$0.4^{-0.21}$	$0.47^{-0.21}$
JRK	$0.44^{-0.16}$	$0.22^{-0.09}$	$0.29^{-0.12}$	$0.14^{-0.28}$	$0.06^{-0.14}$	$0.08^{-0.19}$	$0.36^{-0.07}$	$0.7^{-0.1}$	$0.47^{-0.09}$	$0.19^{-0.15}$	$0.36^{-0.11}$
KLCH	$0.61^{-0.1}$	$0.23^{-0.49}$	$0.33^{-0.38}$	$0.33^{-0.01}$	$0.21^{+0.04}$	$0.26^{+0.04}$	$0.41^{-0.25}$	0.82	$0.55^{-0.18}$	$0.3^{-0.17}$	$0.44^{-0.21}$
GMN	$0.59^{-0.08}$	$0.77^{+0.04}$	$0.66^{-0.04}$	$0.37^{-0.02}$	$0.14^{-0.01}$	$0.2^{-0.01}$	$0.57^{-0.03}$	$0.55^{-0.17}$	$0.56^{-0.1}$	$0.43^{-0.02}$	$0.57^{-0.05}$
MHM	$0.78^{-0.01}$	$0.76^{-0.01}$	$0.77^{-0.01}$	$0.59^{+0.01}$	$0.54^{-0.02}$	$0.56^{-0.01}$	$0.7^{-0.03}$	$0.74^{-0.02}$	$0.72^{-0.02}$	$0.67^{-0.006}$	$0.71^{-0.007}$
GNT	$0.68^{-0.03}$	0.8	$0.73^{-0.02}$	0.55	$0.43^{-0.02}$	$0.48^{-0.02}$	$0.67^{-0.01}$	$0.59^{-0.04}$	$0.62^{-0.03}$	$0.61^{-0.017}$	$0.65^{-0.02}$
Y&C	0.45	1.0	0.62	0.0	0.0	0.0	0.0	0.0	0.0	0.31	0.45
RAE	$0.46^{-0.15}$	$0.98^{+0.37}$	$0.62^{+0.01}$	$0.0^{-0.22}$	$0.0^{-0.01}$	$0.0^{-0.03}$	$0.63^{+0.15}$	$0.05^{-0.67}$	$0.09^{-0.48}$	$0.31^{-0.01}$	$0.46^{-0.08}$
MVRNN	0.45	$0.92^{-0.08}$	$0.6^{-0.02}$	$0.08^{+0.08}$	$0.01^{+0.01}$	$0.01^{+0.01}$	$0.26^{+0.26}$	$0.03^{+0.03}$	$0.06^{+0.06}$	0.31	$0.43^{-0.02}$
RNTN	0.45	$0.93^{+0.11}$	$0.61^{+0.02}$	$0.29^{+0.05}$	$0.01^{-0.05}$	$0.01^{-0.09}$	$0.4^{-0.03}$	$0.07^{-0.1}$	$0.12^{-0.12}$	$0.31^{-0.03}$	$0.45^{-0.01}$
SEV	$0.56^{-0.17}$	$0.79^{+0.05}$	$0.66^{-0.08}$	0.0	0.0	0.0	$0.57^{+0.01}$	$0.57^{-0.27}$	$0.57^{-0.11}$	$0.33^{-0.04}$	$0.56^{-0.08}$
BAZ	$0.65^{-0.17}$	$0.59^{-0.13}$	$0.62^{-0.15}$	0.62	$0.22^{-0.27}$	$0.32^{-0.23}$	$0.5^{-0.18}$	$0.74^{-0.11}$	$0.6^{-0.16}$	$0.47^{-0.19}$	$0.57^{-0.16}$
$LBA^{(1)}$	$0.58^{-0.18}$	$0.77^{-0.07}$	$0.66^{-0.13}$	$0.54^{-0.06}$	$0.53^{-0.03}$	$0.54^{-0.04}$	$0.63^{-0.12}$	$0.37^{-0.31}$	$0.46^{-0.26}$	$0.6^{-0.09}$	$0.58^{-0.15}$
$LBA^{(2)}$	$0.67^{-0.17}$	$0.52^{-0.21}$	$0.59^{-0.19}$	$0.51^{-0.06}$	$0.44^{-0.04}$	$0.47^{-0.06}$	$0.52^{-0.14}$	$0.7^{-0.12}$	$0.6^{-0.13}$	$0.53^{-0.12}$	$0.57^{-0.15}$
SB10k											
HL	$0.41^{-0.08}$	$0.42^{-0.2}$	$0.42^{-0.13}$	$0.24^{-0.03}$	$0.28^{-0.06}$	$0.26^{-0.04}$	$0.66^{-0.07}$	$0.63^{-0.01}$	$0.65^{-0.02}$	$0.34^{-0.08}$	$0.53^{-0.05}$
TBD	$0.41^{-0.07}$	$0.37^{-0.23}$	$0.39^{-0.14}$	$0.21^{-0.03}$	$0.24^{-0.03}$	$0.22^{-0.03}$	$0.65^{-0.07}$	$0.66^{+0.03}$	$0.66^{-0.01}$	$0.31^{-0.08}$	$0.53^{-0.04}$
MST	$0.4^{-0.05}$	$0.32^{-0.17}$	$0.35^{-0.12}$	$0.26^{-0.03}$	$0.3^{-0.05}$	$0.28^{-0.04}$	$0.65^{-0.05}$	$0.68^{-0.04}$	0.67	$0.32^{-0.08}$	$0.54^{-0.03}$
JRK	$0.4^{-0.01}$	$0.42^{-0.03}$	$0.41^{-0.01}$	0.36	0.26	0.3	0.69	$0.72^{-0.03}$	$0.71^{-0.01}$	$0.36^{+0.01}$	$0.59^{-0.006}$
KLCH	$0.42^{+0.03}$	$0.21^{-0.01}$	0.28	$0.25^{-0.09}$	0.13	$0.17^{-0.02}$	0.66	0.86	0.75	$0.23^{-0.005}$	$0.6^{-0.002}$
GMN	$0.48^{-0.17}$	$0.31^{-0.14}$	$0.37^{-0.16}$	$0.27^{-0.11}$	$0.07^{-0.01}$	$0.11^{-0.02}$	$0.69^{-0.03}$	$0.9^{-0.03}$	$0.78^{-0.03}$	$0.24^{-0.09}$	$0.64^{-0.06}$
MHM	$0.67^{-0.04}$	$0.62^{-0.03}$	$0.65^{-0.03}$	$0.59^{-0.08}$	$0.42^{-0.02}$	$0.49^{-0.04}$	0.8	$0.88^{-0.01}$	0.84	$0.56^{-0.002}$	$0.75^{-0.001}$
GNT	$0.42^{-0.25}$	$0.21^{-0.41}$	$0.28^{-0.36}$	$0.25^{-0.19}$	$0.13^{-0.15}$	$0.17^{-0.17}$	$0.66^{-0.12}$	$0.86^{-0.01}$	$0.75^{-0.07}$	$0.22^{-0.2}$	$0.604^{-0.12}$
Y&C	0.0	0.0	0.0	0.0	0.0	0.0	0.62	1.0	0.77	0.0	0.62
RAE	$0.46^{-0.04}$	$0.62^{-0.11}$	$0.53^{-0.06}$	$0.18^{-0.17}$	$0.02^{-0.04}$	$0.03^{-0.07}$	$0.77^{-0.03}$	$0.82^{+0.02}$	$0.79^{-0.01}$	$0.28^{-0.07}$	$0.66^{-0.02}$
MVRNN	0.19	0.01	0.03	0.0	0.0	0.0	0.62	0.97	0.76	0.01	0.61
RNTN	0.0	0.0	0.0	0.0	0.0	0.0	0.62	1.0	0.77	0.0	0.62
SEV	$0.58^{-0.06}$	$0.39^{-0.19}$	$0.47^{-0.14}$	$0.23^{-0.28}$	$0.05^{-0.16}$	$0.08^{-0.22}$	$0.7^{-0.06}$	$0.92^{+0.03}$	$0.8^{-0.02}$	$0.27^{-0.18}$	$0.67^{-0.05}$
BAZ	$0.69^{-0.03}$	$0.54^{-0.16}$	$0.6^{-0.05}$	$0.36^{-0.17}$	$0.49^{+0.16}$	0.41	0.79	$0.79^{-0.12}$	$0.79^{-0.05}$	$0.51^{-0.02}$	$0.69^{-0.06}$
$LBA^{(1)}$	$0.24^{-0.36}$	$0.86^{-0.14}$	$0.38^{-0.28}$	$0.45^{-0.02}$	$0.45^{+0.03}$	$0.45^{+0.01}$	$0.69^{-0.15}$	$0.01^{-0.79}$	$0.02^{-0.8}$	$0.41^{-0.14}$	$0.27^{-0.46}$
$LBA^{(2)}$	$0.74^{-0.02}$	$0.42^{-0.15}$	$0.54^{-0.1}$	$0.62^{+0.07}$	$0.25^{-0.14}$	$0.35^{-0.11}$	$0.73^{-0.06}$	$0.95^{+0.05}$	$0.82^{-0.02}$	$0.45^{-0.1}$	$0.72^{-0.03}$

Table 6.12: Results of CGSA methods without text normalization

HL – Hu and Liu (2004), TBD – Taboada et al. (2011), MST – Musto et al. (2014), JRK – Jurek et al. (2015), KLCH – Kolchyna et al. (2015), GMN – Gamon (2004), MHM – Mohammad et al. (2013), GNT – Günther et al. (2014), Y&C – Yessenalina and Cardie (2011), RAE – Recursive Auto-Encoder (Socher et al., 2011), MVRNN – Matrix-Vector RNN (Socher et al., 2012), RNTN – Recursive Neural-Tensor Network (Socher et al., 2013), SEV – Severyn and Moschitti (2015b), BAZ – Baziotis et al. (2017), $LBA^{(1)}$ – lexicon-based attention with one Bi-LSTM layer, $LBA^{(2)}$ – lexicon-based attention with two Bi-LSTM layers

6.7 Summary and Conclusions

Now the we have reached the end of the chapter, we would like to remind the reader that in this part of the thesis we have made the following findings and contributions:

- we have compared three major families of coarse-grained sentiment analysis methods—lexicon-, machine-learning– and deep-learning–based ones, finding that the last two groups significantly outperform lexicon-driven systems;

- somewhat surprisingly, among all compared lexicon methods, the most simple one—the classifier of Hu and Liu (2004)—produced the best macro- and micro-averaged F_1-results on the PotTS corpus (0.615 and 0.685 respectively) and also yielded the highest macro F_1-measure on the SB10k dataset (0.421). Other systems, however, could have improved their scores if they better handled the negation of polar terms (after switching off the negation component in the method of Musto et al., its macro-F_1 on the PotTS corpus increased to 0.641, surpassing the benchmark of Hu and Liu);

- as expected, the ML-based system of Mohammad et al. (2013)—the winner of the inaugural run of SemEval task in sentiment analysis of Twitter (Nakov et al., 2013)—also surpassed all its competitors from the ML group, achieving highly competitive results: 0.674 macro- and 0.727 micro-F_1 on the PotTS data and 0.564 macro- and 0.752 micro-averaged F_1-measure on the SB10k test set;

- however, as in the previous case, these results could have been improved if the classifier dispensed with character-level and part-of-speech features and used logistic regression instead of SVM;

- a much more varied situation was observed with deep-learning–based systems, which frequently simply degraded to always predicting the majority class for all input tweets, but sometimes yielded extraordinarily good results as it was, for example, the case with our proposed lexicon-based attention system, which attained 0.69 macro-F_1 on the PotTS corpus and 0.55 macro F_1-score on the SB10k dataset (0.73 and 0.75 micro-F_1 respectively), setting a new state of the art for the former collection;

- speaking of word embeddings, we should note that almost all DL-based approaches showed fairly low scores when they used randomly initialized task-specific embeddings, but their results notably improved after switching to pre-trained word2vec vectors, and became even better when we optimized these embeddings with respect to the task at hand and applied the least-squares fallback to approximate the representations of unseen tokens;

- contrary to our initial expectations, we could not overcome the majority class pitfall of DL-based systems after adding more noisily supervised training data, which, in general, only lowered the scores of both ML- and DL-based methods. Since this result contradicts the findings of other authors, we hypothesize that this deterioration is primarily due to the differences in the class distributions between automatically and manually labeled tweets;

- on the other hand, we could see that using more qualitative sentiment lexicons (especially manually curated and dictionary-based ones) resulted in further improvements for the systems which relied on this lexical resource;
- last but not least, we proved the utility of the text normalization step, which brought about significant improvements for all tested methods, as confirmed by our last ablation test.