Proceedings of the GermEval 2017 –
Shared Task on Aspect-based Sentiment in Social Media Customer Feedback
Foreword

In the connected, modern world, customer feedback is a valuable source for insights on the quality of products or services. This feedback allows other customers to benefit from the experiences of others and enables businesses to react on requests, complaints or recommendations. However, the more people use a product or service, the more feedback is generated, which results in the major challenge of analyzing huge amounts of feedback in an efficient, but still meaningful way.

Aspect-based Sentiment Analysis is an important task to analyze customer feedback and a growing number of Shared Tasks exist for various languages. However, these tasks lack large-scale German datasets. Thus, we present a shared task on automatically analyzing customer reviews about “Deutsche Bahn” – the German public train operator with about two billion passengers each year. We have annotated more than 26,000 documents and present four sub-tasks that represent a complete classification pipeline (relevance, sentiment, aspect classification, opinion target extraction).

The results indicate that the public transport domain offers challenging tasks. E.g., the large number of aspects – in combination with an almost Zipfian label distribution of real user feedback – leads to label bias problems and creates strong baselines. We observe that the usage of extensive preprocessing, large sentiment lexicons, and the connection of neural and more traditional classifiers are advantageous strategies for the formulated tasks. The Shared Task is a first step in sentiment analysis for this domain.

For the GermEval 2017 Shared Task, we received 8 submissions. One submission was withdrawn from the proceedings. The dataset and the proceedings are available from the task website (https://sites.google.com/view/germeval2017-absa/). It also contains the presentation slides from the participants and the individual prediction labels from the participating systems.

Berlin, September 2017

The organizing committee

Organizers:

Michael Wojatzki (Universität Duisburg-Essen)
Eugen Ruppert (Universität Hamburg)
Torsten Zesch (Universität Duisburg-Essen)
Chris Biemann (Universität Hamburg)
# Table of Contents

**GermEval 2017: Shared Task on Aspect-based Sentiment in Social Media Customer Feedback**  
Michael Wojatzki, Eugen Ruppert, Sarah Holschneider, Torsten Zesch and Chris Biemann  . . . . . . 1

**h_da Participation at Germeval Subtask B: Document-level Polarity**  
Karen Schulz, Margot Mieskes and Christoph Becker ........................................... 13

**HU-HHU at GermEval-2017 Sub-task B: Lexicon-Based Deep Learning for Contextual Sentiment Analysis**  
Behzad Naderalvojoud, Behrang Qasemizadeh and Laura Kallmeyer .................................. 18

**UKP TU-DA at GermEval 2017: Deep Learning for Aspect Based Sentiment Detection**  
Ji-Ung Lee, Steffen Eger, Johannes Daxenberger and Iryna Gurevych  . . . . . . . . . . . . . . . . . 22

**Fasttext and Gradient Boosted Trees at GermEval-2017 on Relevance Classification and Document-level Polarity**  
Leonard Hövelmann and Christoph M. Friedrich .......................................................... 30

**GermEval 2017 : Sequence based Models for Customer Feedback Analysis**  
Pruthwik Mishra, Vandan Mujadia and Soujanya Lanka ........................................... 36

**DIDS IUCL: Investigating Feature Selection and Oversampling for GermEval2017**  
Zeeshan Ali Sayyed, Daniel Dakota and Sandra Kübler ........................................... 43

Uladzimir Sidarenka ................................................................. 49

**LT-ABSA: An Extensible Open-Source System for Document-Level and Aspect-Based Sentiment Analysis**  
Eugen Ruppert, Abhishek Kumar and Chris Biemann .................................................. 55
GermEval 2017: Shared Task on Aspect-based Sentiment
in Social Media Customer Feedback

Michael Wojatzki† and Eugen Ruppert‡ and Sarah Holschneider⋄ and Torsten Zesch† and Chris Biemann‡

†Language Technology Lab CompSci and Appl. CogSci Universität Duisburg-Essen
http://www.ltl.uni-due.de/

‡Technische Universität Darmstadt Computer Science Department Universität Hamburg
http://www.ltu.darmstadt.de http://lt.informatik.uni-hamburg.de

Abstract

This paper describes the GermEval 2017 shared task on Aspect-Based Sentiment Analysis that consists of four subtasks: relevance, document-level sentiment polarity, aspect-level polarity ad opinion target extraction. System performance is measured on two evaluation sets – one from the same time period as the training and development set, and a second one, which contains data from a later time period. We describe the subtasks and the data in detail and provide the shared task results. Overall, the shared task attracted over 50 system runs from 8 teams.

1 Introduction

In a connected, modern world, customer feedback is a valuable source for insights on the quality of products or services. This feedback allows other customers to benefit from the experiences of others and enables businesses to react on requests, complaints or recommendations. However, the more people use a product or service, the more feedback is generated, which results in the major challenge of analyzing huge amounts of feedback in an efficient, but still meaningful way.

Recently, shared tasks on Sentiment Analysis have been organized regularly, the most popular are the shared tasks in the SemEval framework (Pontiki et al., 2015; Pontiki et al., 2016). And even though the number of domains and languages is growing with each iteration, there has not existed a large public sentiment analysis dataset for German until now.

To fill this gap, we conducted a shared task1 on automatically analyzing customer reviews and news about “Deutsche Bahn” – the major German public train operator, with about two billion passengers each year. This is the first shared task on German sentiment analysis that provides a large annotated dataset for training and evaluating machine learning approaches. Furthermore, it features one of the largest datasets for sentiment analysis overall, containing annotations on almost 28,000 short documents, more than 10 times of the training instances in the largest set to date (from SemEval-2016, task 5 ‘Arabic Hotels’).

2 Task Description

The shared task features four subtasks, which can be tackled individually. They are aimed at realizing a full classification pipeline when dealing with web data from various heterogeneous sources. First, in Subtask A, the goal is to determine whether a review is relevant to our topic. In real life scenarios this task is necessary to filter irrelevant documents that are a by-catch of the method of collecting the data. Second, Subtask B is about inferring a customer’s overall evaluation of the Deutsche Bahn based on the given document. Here, we support a use-case in which e.g. a manager is interested how well or badly the offered services are perceived overall. Third, Subtask C addresses a more fine-grained level and aims at finding the particular kind of service, called aspect, which is referred to positively or negatively. Finally, in Subtask D the task is to identify the actual expressions that verbalize the evaluations covered in Subtask C, commonly known as opinion target expression (OTE) identification.

2.1 Subtask A: Relevance Classification

The first subtask is used to filter incoming documents so that only the relevant and interesting ones are processed further. The term Bahn can refer to many different things in German: the rails, the train, any track or anything that can be laid in straight
In Subtask A, the documents have to be labeled in a binary classification task as relevant (true) or irrelevant (false) for Deutsche Bahn. Below is a relevant document about bad behavior in a train, and an irrelevant document about stock exchange developments.

true Ehrlich die männer in Der Bahn haben keine manieren? (Seriously, the men in those trains have no manners!)

false Aus der Presseschau: Japanese S-Bahn wird mit Spiegelwaggons ‘unsichtbar’ (Review: Japanese urban railway becomes ‘invisible’ thanks to reflecting wagons)

2.2 Subtask B: Document-level Polarity

In Subtask B, systems have to identify, whether the customer evaluates the service of the railway company, be it e.g. travel experience, timetables or customer communication as positive, negative, or neutral. During data acquisition, annotators provided more complex aspect-level annotations as used in Subtasks C and D. Document level sentiment polarity in Subtask B is computed from the individual aspect polarities in the document: If there is a mixture between neutral and positive/negative, the documents are classified as positive/negative. If there are two opposing polarities (positive and negative), the overall sentiment is set to neutral.

2.3 Subtask C: Aspect-level Polarity

For Subtask C, participants are asked to identify all aspects in the document. Each aspect should be labeled with the appropriate polarity label. Since, in the annotations, it was possible to label multiple tokens with the same aspect, multiple mentions of the same aspect are possible. The example below shows a mixed sentiment in a document that is presented as a dialogue.

The positive aspect is the end of a strike – Streik beendet. The negative aspect in this document are the tickets, which are getting more expensive – die Tickets teurer. Thus, in the given example, the task is to identify the aspects (and their polarity) in the following way: Ticketkauf#Haupt:negative, Allgemein#Haupt:positive.

2.4 Subtask D: Opinion Target Extraction

For the last subtask, participants should identify the linguistic expressions that are used to express the aspect-based sentiment (Subtask C). The opinion target expression is defined by its starting and ending offsets. For human readability, the target terms are also present in the data as well.

An example is given in Listing 1. In this document, the task is to identify the target expression fährt nicht (does not drive/go), which is an indication of an irregularity in the operating schedule.

The aspect classification was provided by the data analysis from Deutsche Bahn and was refined during the annotation process.

The data set is available in both TSV and XML formats (see Section 3.4), Subtask D can only be done using the XML format, as the spans of the opinion target expression are not available in the
document-based TSV format. For more detail, see the next section.

3 Dataset

3.1 Data Collection

The data was crawled from the Internet on a daily basis with a list of query terms. We filtered for German documents and focused on social media, microblogs, news, and Q&A sites. Besides the document text, meta information like URL, date, and language was collected as well.

In the project context, we received more than 2.5 million documents overall, spanning a whole year (May 2015–June 2016), so that we could capture all possible seasonal problems (holidays, heat, cold, strikes) as well as daily problems such as delays, canceled trains, or unclear information at the train stations. From this large amount of documents, we sampled from each month approximately 1,500 documents for annotation. Since the word-list-based relevance filtering is very coarse and a lot of irrelevant documents were present in the initial samples, e.g. questions about the orbit of the moon (Mondumlaußbahn, lunar orbit) or mentions of air draft (zugig, drafty), we trained a baseline SVM classifier to perform pre-filtering and increase the number of relevant and interesting documents per split. The annotated data is used for the training, development, as well as for a synchronic test set.

Additionally, to test the robustness of the participating systems, we created a diachronic test set, which was (pre-)processed and annotated in the same manner, using data from November 2016 to January 2017.

3.2 Annotation

For annotation, we used WebAnno (de Castilho et al., 2016). Annotators were asked to perform the full annotation of every document assigned to them. To keep the individual tasks manageable, we split the annotation tasks into chunks of 100 short documents, which could be completed in 1–2 hours by an annotator.

The annotation task consisted of first labeling the document relevance. For relevant documents, the annotators had to identify the aspect targets (spans of single or multiple tokens) and label them with one of 19 aspects and, if identifiable, with one of overall 43 sub-aspects.

Relevant documents that did not contain a clear aspect target expression could also be assigned a document-level aspect annotation. The polarity words for each aspect target were annotated as well. If they were not part of the OTE (as e.g. Verspätung – delay, which is inherently negative), they were connected with the aspect-bearing word using a relation arc. These annotations have not been distributed as part of this challenge, but will be made available afterwards. Expressions of the same aspect were also connected from left to right.

The annotation team consisted of six trained student annotators and a supervisor/curator. Every document was annotated by two annotators in differing pairings. The curator checked the documents for diverging annotations and decided on the correct one using WebAnno’s curation interface. Furthermore, she was also able to add new annotations, in case the others missed some expressions. In weekly feedback sessions, the team talked about new problems and added the results to the annotation guidelines. This led to consistent improvements of inter-annotator agreement over time, see Table 3 and Figure 1. The overall lower agreements for the Relevance classifications are due to the difficulty of deciding between irrelevant documents and documents without explicit sentiments.

2The German annotation guidelines are available at: http://ltdata1.informatik.uni-hamburg.de/germeval2017/Guidelines_DB_v4.pdf
Figure 1: Development of the inter-annotator agreement over time

<table>
<thead>
<tr>
<th>Date</th>
<th>07.2016</th>
<th>10.2016</th>
<th>01.2017</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relevance</td>
<td>0.26–0.51</td>
<td>0.39–0.74</td>
<td>0.51–0.76</td>
</tr>
<tr>
<td>Polarity</td>
<td>0.35–0.79</td>
<td>0.45–0.97</td>
<td>0.90–1.00</td>
</tr>
<tr>
<td>Aspect</td>
<td>0.42–0.70</td>
<td>0.44–0.93</td>
<td>0.79–1.00</td>
</tr>
<tr>
<td>Sub-Aspect</td>
<td>0.35–0.65</td>
<td>0.37–0.87</td>
<td>0.63–1.00</td>
</tr>
</tbody>
</table>

Table 3: Development of the inter-annotator agreement ranges (Cohen’s kappa)

3.3 Splits

We obtained about 26,000 annotated documents for the main dataset and about 1,800 documents for the diachronic dataset. We split the main dataset into a training, development and test set using a random 80%/10%/10% split. The number of documents for each split is shown in Table 4. The dataset can be downloaded from: http://ltdata1.informatik.uni-hamburg.de/germeval2017/

Tables 5–7 show the label distributions for the subtasks. There is always a clear majority class, which leads to strong baselines. This is especially apparent for Subtask C (Table 7), where the most frequent label Allgemein (General) is almost 10 times as frequent as the second frequent label.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>true</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training</td>
<td>16,201</td>
<td>3,231</td>
</tr>
<tr>
<td>Development</td>
<td>1,931</td>
<td>438</td>
</tr>
<tr>
<td>Test_syn</td>
<td>2,095</td>
<td>471</td>
</tr>
<tr>
<td>Test_dia</td>
<td>1,547</td>
<td>295</td>
</tr>
</tbody>
</table>

Table 4: Number of documents in data splits

<table>
<thead>
<tr>
<th>Dataset</th>
<th>negative</th>
<th>neutral</th>
<th>positive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training</td>
<td>5,045</td>
<td>13,208</td>
<td>1,179</td>
</tr>
<tr>
<td>Development</td>
<td>589</td>
<td>1,632</td>
<td>148</td>
</tr>
<tr>
<td>Test_syn</td>
<td>780</td>
<td>1,681</td>
<td>105</td>
</tr>
<tr>
<td>Test_dia</td>
<td>497</td>
<td>1,237</td>
<td>108</td>
</tr>
</tbody>
</table>

Table 5: Relevance Distribution in Subtask A data

Table 6: Sentiment Distribution in Subtask B data

3.4 Formats

We utilize an XML format that is similar to the format used in SemEval-2016 task on ABSA (Task 5) (Pontiki et al., 2016). Each Document element contains the original URL as the document id and
Table 7: Distribution of top-frequent aspects (aspects partly shortened) in Subtask C data

<table>
<thead>
<tr>
<th>Training</th>
<th>Development</th>
<th>Test_syn</th>
<th>Test_dia</th>
</tr>
</thead>
<tbody>
<tr>
<td>11,191 Allgemein</td>
<td>1,363 Allgemein</td>
<td>1,351 Allgemein</td>
<td>1,008 Allgemein</td>
</tr>
<tr>
<td>1,240 Zugfahrt</td>
<td>140 Zugfahrt</td>
<td>178 Sonstige...</td>
<td>144 Zugfahrt</td>
</tr>
<tr>
<td>1,007 Sonstige_Unregelmäßigkeiten</td>
<td>108 Atmosphäre</td>
<td>160 Zugfahrt</td>
<td>138 Sonstige...</td>
</tr>
<tr>
<td>819 Atmosphäre</td>
<td>102 Sonstige...</td>
<td>112 Atmosphäre</td>
<td>72 Connectivity</td>
</tr>
<tr>
<td>417 Ticketkauf</td>
<td>51 Ticketkauf</td>
<td>75 Ticketkauf</td>
<td>42 Atmosphäre</td>
</tr>
<tr>
<td>296 Service_und_Kundenbetreuung</td>
<td>37 Sicherheit</td>
<td>51 Sicherheit</td>
<td>29 Ticketkauf</td>
</tr>
<tr>
<td>278 Sicherheit</td>
<td>29 Service...</td>
<td>42 Service...</td>
<td>27 Sicherheit</td>
</tr>
<tr>
<td>224 Connectivity</td>
<td>22 Connectivity</td>
<td>31 Informat...</td>
<td>21 Informat...</td>
</tr>
<tr>
<td>193 Informationen</td>
<td>19 Auslastung...</td>
<td>27 Connectivity</td>
<td>18 Service...</td>
</tr>
<tr>
<td>158 Auslastung_und_Platzangebot</td>
<td>14 DB_App_und_Website</td>
<td>22 Auslastung...</td>
<td>15 Auslastung...</td>
</tr>
</tbody>
</table>

∑ Rest 377 ... 45 ... 46 ... 33 ...

# Aspects 16,200 1,930 2,095 1,547

# non-Null Asp. 12,139 1,380 2,162 1,163

The extracted untokenized document text. Furthermore, the relevance and the document polarity are annotated as well. For relevant documents, the opinion target expressions (OTE) are grouped as Opinions. Each Opinion contains the token offsets for the OTE, its aspect and the sentiment polarity. Two examples are given in Listing 2. The first one has identifiable OTEs, while the second one – although relevant – does not provide an explicit opinion target expression.

To increase participation and lower the entry boundary, we also provide an TSV format for document-level annotation in order to enable straightforward use with any document classifier. The TSV format contains the following tab-separated fields:

- document id (URL)
- document text
- relevance (true or false)
- document-level polarity, neutral for irrelevant documents
- aspects with polarities; several mentions are possible, empty for irrelevant documents
  Example: Atmosphäre#Haupt:neutral Atmosphäre#Lautstärke:negative

Since there are only document-level labels for the TSV format, Subtask D is not evaluated for TSV submissions.

4 Evaluation Measures and Baselines

We evaluate the system predictions using a micro-averaged F1 score. This metric is well-suited for datasets with a clear majority class because each instance is weighted the same as every other one.

For Subtasks A and B (relevance and document-level polarity), we only report the F1 score. For the aspect identification (Subtask C), we report scores for the aspect identification itself, as well as the combination of aspect and sentiment, as it can differ between several aspects in a document.

Opinion target expression matching is evaluated in an exact setting, where the token offsets have to match exactly, and a less strict setting, which considers overlaps and partial matches as correct. In detail, we consider an expression a match if the span is +/- one token of the gold data.

Majority Class Baseline The majority class baseline (MCB) yields already a quite good performance, since Subtasks A, B and C have a clear majority class. Thus, the majority class is a strong prior or fallback alternative for instances without much evidence for the other classes. For Subtask C we assign exactly one opinion with the aspect Allgemein and the sentiment neutral. Since Subtask D is a sequence tagging task, there is no meaningful majority class baseline.

Baseline System We provided a baseline system that uses machine learning with a basic feature set to show the improvements put forth by the participating system. It uses a linear SVM classifier for Subtasks A, B and C and a CRF classifier for the OTEs in Subtask D, both with a minimal set of standard features. The baseline system is available for the participants for initial evaluation and a possible weakly-informed classifier in an ensemble learning setting. The system is available under the Apache Software License 2.0 at: https://github.com/uhh-lt/
5 Participation

Overall, 8 teams participated in the shared task. All of them participated in Subtask B and 5 of them in Subtask A. Only Lee et al. (2017) and Mishra et al. (2017) have participated in Subtasks C and D. Table 8 gives an overview of which team participated in which subtask.

5.1 Participant’s Approaches

Across all subtasks, the participants have applied a large variety of approaches. However, we can identify trends and commonalities between the teams, which will be discussed in more detail below. For a detailed description of the approaches, we refer to the referenced papers.

Preprocessing

Although some teams have used off-the-shelf tokenizers, such as Schulz et al. (2017) who used the opennlp maxent tokenizer, most of the teams relied on their own implementations. These tokenizers were often combined with large sets of rules that cover social media specific language phenomena such as emoticons, URLs, or repeated punctuation (Sayyed et al., 2017; Sidarenka, 2017; Mishra et al., 2017; Hövelmann and Friedrich, 2017). It would have been possible to use tokenizers from the 2016 EMPIRIST task, e.g. (Remus et al., 2016). Moreover, one team (Hövelmann and Friedrich, 2017) further normalized the data by using an off-the-shelf spell checker and rules to replace e.g. numbers, dates, and URLs with a special token.

Besides a tokenizer, many recent neural classifiers do not require deeper preprocessing. Never-
nevertheless, some of the participants used lemmatizers, chunkers, and part-of-speech taggers (Sidarenka, 2017; Schulz et al., 2017; Naderalvojoud et al., 2017), relying on the TreeTagger by Schmid (1994) or on the Stanford CoreNLP library (Manning et al., 2014).

To compensate for imbalances in the class distribution, two teams have used sampling techniques (Sayyed et al. (2017) and UH-HHU-G) – namely adaptive synthetic sampling (He et al., 2008) and synthetic minority over-sampling (Chawla et al., 2002).

**Sentiment Lexicons**

Most teams used or experimented with some form of word polarity resources. Two teams (Schulz et al., 2017; Sidarenka, 2017) relied on SentiWS (Remus et al., 2010). The resource was also considered but not included in the actual submissions of Hövelmann and Friedrich (2017). Two teams (Schulz et al., 2017; Mishra et al., 2017) have used the lexicon created by Wattinger (2010). Other similarly used resources include the Zurich Polarity List (Clematide and Klenner, 2010) or the LWIC tool (Tausczik and Pennebaker, 2010).

In addition to the use of pre-calculated or manual resources, some teams also created their own lexicons. For instance, Naderalvojoud et al. (2017) created a sense based sentiment lexicon from a large subtitle corpus. Sidarenka (2017) created several lexicons e.g. based on other pre-existing dictionaries and using a German Twitter snapshot.

**Dense Word Vectors**

In addition to word polarity, several teams made use of dense word vectors (also known as word embeddings) and thus integrated distributional semantic word information in their systems. Mishra et al. (2017) trained dense word vectors on large corpus of parliament speeches using GloVe (Pennington et al., 2014). Lee et al. (2017) used word2vec (Mikolov et al., 2013) trained word embeddings on Wikipedia. They also trained sentence vectors on the same data and experimented with German-English bilingual embeddings. Finally, some of the teams relied on fastText (Bojanowski et al., 2017) that makes use of sub-word information to create word vectors, addressing phenomena such as German single-token compounding.

**Classifiers**

When analyzing the classification algorithms utilized by the participants, we identify three major strands. The first strand are approaches that use engineered features to represent the data together with more traditional classification algorithms. The second strand translates the training data in sequences of vectors and feeds them into neural networks. Third, there are ensemble approaches that orchestrate several neural and/or non-neural approaches.

Within the non-neural strand we observe the usage of SVMs (Sidarenka, 2017), CRFs (Mishra et al., 2017; Lee et al., 2017), and threshold based classification (Schulz et al., 2017). Approaches of the neural strand used several different neural network architectures. Most dominant is the usage of recurrent neural networks that contain long-short-term-memory (LSTM) units (UH-HHU-G). In particular, many teams used biLSTM - a variant of LSTMs in which both preceding and following context is considered (Sidarenka, 2017; Mishra et al., 2017; Naderalvojoud et al., 2017; Lee et al., 2017). Other used architectures include convolution layers (UH-HHU-G) and other forms of structured or multi-layered perceptrons (Mishra et al., 2017). Within the ensemble approaches there is an approach of orchestrating several neural networks (Lee et al., 2017), one that combines LSTM and SVM (Sidarenka, 2017), one that uses fastText (Hövelmann and Friedrich, 2017) and two approaches that rely on gradient boosted trees (Hövel-

---

4Submission withdrawn after reviewing

---

<table>
<thead>
<tr>
<th>Team reference</th>
<th>Team name</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Schulz et al. (2017)</td>
<td>hda</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>UH-HHU-G</td>
<td>UH-HHU-G</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lee et al. (2017)</td>
<td>UKP_Lab_TUDA</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Mishra et al. (2017)</td>
<td>im+sing</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Sayyed et al. (2017)</td>
<td>IDS_IULC</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sidarenka (2017)</td>
<td>PotTS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Naderalvojoud et al. (2017)</td>
<td>HU-HHU</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hövelmann and Friedrich (2017)</td>
<td>fhdo</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 8: Teams and subtask participation

---

7
mann and Friedrich, 2017; Sayyed et al., 2017).

6 Evaluation Results

As expected, we observe an increasing difficulty between the subtasks in alphabetical order. This means Subtask A is solved better than B, B solved better than C and C is solved better than D. Interestingly, for all tasks we only see small differences between synchronic and diachronic test sets. From this, we can conclude that either all models are robust against temporary fluctuation, or the distributions in this data do not change at a very high speed. Furthermore, both the majority class baseline and our simple baseline system are quite competitive in all tasks. The detailed results of the subtasks are discussed below.

6.1 Subtask A - Relevance

Most of the teams that participated in Subtask A have beaten the majority class baseline and the baseline system. Table 9 gives an overview of the results. Note that the majority class baseline (0.816) and baseline system (0.852) in this subtask are quite strong. The best system by Sayyed et al. (2017) surpassed our baseline system by 0.05 percent point by using gradient boosted trees and feature selection to obtain the predictions. The second-best team (Hövelmann and Friedrich, 2017) used fastText and applied extensive preprocessing. In future research, it seems worthwhile to examine how these strategies contribute to each system. In addition, we note that the neural approaches of Mishra et al. (2017) and Lee et al. (2017) are almost en par (≈ −0.02).

6.2 Subtask B - Document-level Polarity

Similar to Subtask A, we also observe strong baselines in Subtask B, yet that most participants surpass them. Table 10 shows the results. Performance among the top three teams is highly similar. This is particularly interesting as the top three teams Naderalvojoud et al. (2017) [0.749], Hövelmann and Friedrich (2017) [0.748] and Sidarenka (2017) [0.745] have all followed completely different approaches. Naderalvojoud et al. (2017) [0.749] made use of a large lexicon that was combined with a neural network. As already described above, Hövelmann and Friedrich (2017) [0.748] used fastText and extensive preprocessing of the data, whereas Sidarenka (2017) relied on a biLSTM/SVM ensemble. The more or less pure neural approaches of Lee et al. (2017), Sidarenka (2017), and UH-HHU-G yield a slightly worse performance, but still outperform our simple baseline system. Overall, we do not observe large difference on the synchronic versus the diachronic test set, however, most systems marginally lose performance on the diachronic data.

6.3 Subtask C - Aspect-level Polarity

Table 11 shows the performance of the two teams that participated in the aspect-based subtask. Only (Lee et al., 2017) could outperform both provided baselines on the synchronic data. However, the improvements of 0.001 for aspect classification and 0.03 for aspect and sentiment classification are only slight. Surprisingly, on the diachronic data both teams could neither significantly outperform the baseline system nor the majority class baseline (Allgemein:neutral). Interestingly, we observe a increased performance for all submitted runs for the diachronic data.

6.4 Subtask D - Target Extraction

The same teams that worked on Subtask C also participated in Subtask D. Both teams relied on neural network approaches and outperformed both baselines. While the structured perceptron of Mishra et al. (2017) achieved the best results for the exact metric, the combination of LSTM and CRF by Lee et al. (2017) gained the – by far – best results for the overlap metric. In Table 12 we report the results. As expected, the results of the overlap metric are better than those of the exact metric, as the exact metric is more strict. Similar to subtask C, we can conclude that the diachronic data can be classified more easily in both metrics.

7 Related Work

First of all, our shared task is related to shared tasks on aspect-based sentiment analysis that were conducted within the international workshop on semantic evaluation (SemEval) (Pontiki et al., 2014; Pontiki et al., 2015; Pontiki et al., 2016). However, we here focus exclusively on German but target a larger, monolingual data set. We also relate to previous German shared tasks on aspect-based sentiment analysis such as Ruppenhofer et al. (2016). In contrast to this work, we are pursuing an annotation scheme that is inspired by the needs of an industrial customer as opposed to linguistic considerations.
As we are examining directed opinions, we also relate to shared tasks that were conducted on automatically detecting stance from social media data. Stance is defined as being in favor or against a given target, which can be a politician, a political assertion or any controversial issue. Stance detection has been addressed by a couple of recent shared tasks – namely SemEval 2016 task 6 (Mohammad et al., 2016), NLPCCC Task 4 (Xu et al., 2016) or IBEREVAL 2017 (Taulé et al., 2017). Similar to them, we find that state-of-the-art methods still have a long way to go to solve the problem and that, in contrast to other domains and tasks, neural networks are not clearly superior and often inferior to more traditional rule-based or feature engineering approaches.

### 8 Conclusions

In this paper, we describe a shared task on aspect-based sentiment analysis in social media customer feedback. Our shared task includes four subtasks, in which the participants had to detect A) whether feedback is relevant to the given topic Deutsche Bahn, B) which overall sentiment is expressed by a review, C) what aspects are evaluated, and D) what linguistic expressions are used to express these aspects. We provide an annotated data set of almost 28,000 messages from several social media sources. Thereby our dataset represents the largest set of German sentiment annotated reviews.

The shared task attracted a high variance of approaches from 8 different teams. We observe that the usage of gradient boosted trees, large sentiment lexicons, and the connection of neural and more traditional classifiers are advantageous strategies.

---

### Table 9: Results for Subtask A on relevance detection.

<table>
<thead>
<tr>
<th>Team</th>
<th>Run</th>
<th>synchronic</th>
<th>diachronic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sayyed et al. (2017)</td>
<td>xgboost</td>
<td>.903</td>
<td>.906</td>
</tr>
<tr>
<td>Hövelmann and Friedrich (2017)</td>
<td>fasttext</td>
<td>.899</td>
<td>.897</td>
</tr>
<tr>
<td>Mishra et al. (2017)</td>
<td>bilLSTM structured perceptron</td>
<td>.879</td>
<td>.870</td>
</tr>
<tr>
<td>Lee et al. (2017)</td>
<td>stacked learner CCA SIF embedding</td>
<td>.873</td>
<td>.881</td>
</tr>
<tr>
<td>Hövelmann and Friedrich (2017)</td>
<td>gbt_bow</td>
<td>.863</td>
<td>.856</td>
</tr>
<tr>
<td><strong>organizers</strong></td>
<td>baseline system</td>
<td>.852</td>
<td>.868</td>
</tr>
<tr>
<td>UH-HHU-G</td>
<td>ridge classifier char fourgram</td>
<td>.835</td>
<td>.849</td>
</tr>
<tr>
<td>UH-HHU-G</td>
<td>linear SVC I2 char fivegram</td>
<td>.824</td>
<td>.837</td>
</tr>
<tr>
<td>UH-HHU-G</td>
<td>passive-aggressive char fivegram</td>
<td>.827</td>
<td>.850</td>
</tr>
<tr>
<td>UH-HHU-G</td>
<td>linear SVC I2 trigram</td>
<td>.816</td>
<td>.839</td>
</tr>
<tr>
<td><strong>organizers</strong></td>
<td>majority class baseline</td>
<td>.816</td>
<td>.839</td>
</tr>
<tr>
<td>UH-HHU-G</td>
<td>gru mt</td>
<td>.816</td>
<td>.840</td>
</tr>
<tr>
<td>UH-HHU-G</td>
<td>cnn gru sent mt</td>
<td>.810</td>
<td>.839</td>
</tr>
<tr>
<td>Hövelmann and Friedrich (2017)</td>
<td>ensemble</td>
<td>.734</td>
<td>.160</td>
</tr>
</tbody>
</table>

### Table 10: Results for Subtask B on sentiment detection.

<table>
<thead>
<tr>
<th>Team</th>
<th>Run</th>
<th>synchronic</th>
<th>diachronic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Naderalvojoud et al. (2017)</td>
<td>SWN2-RNN</td>
<td>.749</td>
<td>.736</td>
</tr>
<tr>
<td>Hövelmann and Friedrich (2017)</td>
<td>fasttext</td>
<td>.748</td>
<td>.742</td>
</tr>
<tr>
<td>Sidarenka (2017)</td>
<td>biLSTM-svm</td>
<td>.745</td>
<td>.718</td>
</tr>
<tr>
<td>Naderalvojoud et al. (2017)</td>
<td>SWN1-RNN</td>
<td>.737</td>
<td>.736</td>
</tr>
<tr>
<td>Sayyed et al. (2017)</td>
<td>xgboost</td>
<td>.733</td>
<td>.750</td>
</tr>
<tr>
<td>Sidarenka (2017)</td>
<td>biLSTM</td>
<td>.727</td>
<td>.704</td>
</tr>
<tr>
<td>Lee et al. (2017)</td>
<td>stacked learner CCA SIF embedding</td>
<td>.722</td>
<td>.724</td>
</tr>
<tr>
<td>Hövelmann and Friedrich (2017)</td>
<td>gbt_bow</td>
<td>.714</td>
<td>.714</td>
</tr>
<tr>
<td>Hövelmann and Friedrich (2017)</td>
<td>ensemble</td>
<td>.710</td>
<td>.725</td>
</tr>
<tr>
<td>UH-HHU-G</td>
<td>ridge classifier char fourgram</td>
<td>.692</td>
<td>.691</td>
</tr>
<tr>
<td>Mishra et al. (2017)</td>
<td>bilLSTM structured perceptron</td>
<td>.685</td>
<td>.675</td>
</tr>
<tr>
<td>UH-HHU-G</td>
<td>linear SVC I2 char fivegram</td>
<td>.680</td>
<td>.692</td>
</tr>
<tr>
<td><strong>organizers</strong></td>
<td>baseline system</td>
<td>.667</td>
<td>.694</td>
</tr>
<tr>
<td>UH-HHU-G</td>
<td>linear SVC I2 trigram</td>
<td>.663</td>
<td>.702</td>
</tr>
<tr>
<td><strong>organizers</strong></td>
<td>majority class baseline</td>
<td>.656</td>
<td>.672</td>
</tr>
<tr>
<td>UH-HHU-G</td>
<td>gru mt</td>
<td>.656</td>
<td>.672</td>
</tr>
<tr>
<td>UH-HHU-G</td>
<td>cnn gru sent mt</td>
<td>.644</td>
<td>.668</td>
</tr>
<tr>
<td>Schulz et al. (2017)</td>
<td>Passive-Aggressive char fivegram</td>
<td>.612</td>
<td>.616</td>
</tr>
<tr>
<td>UH-HHU-G</td>
<td></td>
<td>.575</td>
<td>.676</td>
</tr>
</tbody>
</table>
for the formulated tasks. Nevertheless, our simple baseline classifier is highly competitive across all tasks. We will release the annotated dataset as part of this task. This will hopefully strengthen the research on German sentiment and social media analysis.

Acknowledgments

We would like to thank Axel Schulz and Maria Pelevina from the Deutsche Bahn Fernverkehr AG for their collaboration in the project ABSA-DB: Aspect-based Sentiment Analysis for DB Products and Services as part of the Innovation Alliance DB & TU Darmstadt. We would also like to thank Ji-Ung Lee for his helpful feedback. In addition, this work was supported by the Deutsche Forschungsgemeinschaft (DFG) under grant No. GRK 2167, Research Training Group “User-Centred Social Media”. Finally, we thank the Interest Group on German Sentiment Analysis (IGGSA) for their endorsement.

References


Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In Association for Computational Linguistics (ACL) System Demonstrations, pages 55–60, Baltimore, MD, USA.


Josef Ruppenhofer, Julia Maria Straß, and Michael Wiegand. 2016. Overview of the IGGSA 2016 Shared Task on Source and Target Extraction from Political Speeches. In Bochumer Linguistische Arbeitsberichte, pages 1–9, Bochum, Germany.


h_da Participation at Germeval Subtask B: Document-level Polarity
Karen Schulz, Margot Mieskes*, Christoph Becker*
University of Applied Sciences Darmstadt
Karen.Schulz@stud.h-da.de, *Firstname.Lastname@h-da.de
University of Applied Sciences Darmstadt

Abstract
This paper describes our participation in subtask B of the Germeval 2017 shared task on Sentiment Polarity Detection in Social Media Customer Feedback. The task asks to classify the provided comments with respect to “Deutsche Bahn” and “travelling”. Our system is based on lexical resources, which are combined to determine the polarity for each sentence. The polarity of the individual sentences is used to classify customer reviews.

1 Introduction
This paper describes our participation in the Germeval 2017 subtask B, which deals with document level polarity on customer reviews in the context of the german railway company (Deutsche Bahn). Originally, our sentiment analysis method was created for English Tweets from the financial domain in R. Therefore, in order to participate in the task at hand, we had to move our current system from one domain (financial), one genre (Twitter) and one language (English) to another domain (travelling by train), another genre (customer reviews) and another language (German).

2 Related Work
Reviewing the available literature on sentiment analysis is beyond the scope of this paper. For an extensive overview on the topic we refer to (Liu and Zhang, 2012) or to (Cambria et al., 2017). An overview on various applications that have been studied in the domain of sentiment analysis can be found for example in Lak and Turetken (2017). Taboada et al. (2011) pointed out, that rule-based systems are more robust to domain changes, which is one of the key issues in our setup.

The majority of work on sentiment analysis has been done on English, but very little on other languages such as German (see for example Denecke (2008)). In some cases, where other languages were analysed, the authors made use of automatic translation tools. One example is Tumasjan et al. (2010), who analysed tweets. As we deal with German data in this case, we focus on previous work using German data directly.

Waltinger (2010) presented a lexicon for German sentiment analysis, which was compiled by translating existing English dictionaries automatically and manually post-process the results. The final lexicon has over 3,000 positive features, almost 6,000 negative features and over 1,000 neutral features. This lexicon was tested on 5-star customer reviews, where 1 and 2 star reviews were collapsed to negative reviews, 4 and 5 star reviews were grouped to positive reviews and 3 star reviews were considered neutral. Comparing positive vs. negative reviews or 1 vs. 5 star reviews using an SVM the authors achieved an F1-average score of .803 to .876.

Remus et al. (2010) also presented a sentiment lexicon, which contains almost 2,000 positive and negative words each, excluding inflections. It is also based on automatic translation and manual revision. Additionally, the authors used co-occurrence analysis and a collocation lexicon. The evaluation is based on individual words and achieves an overall F1-measure of .84. However, the authors observed that positive words achieved better results than negative words.

Momtazi (2012) worked on sentiment analysis of German sentences from various Social Media channels. Her rule-based approach used the counts of positive and negative words. She distinguished between binary classifications, where the majority of words determined the final results, while in a more fine-grained scenario the frequency of the sentiment-bearing words is taken into account as well. Additionally, so-called booster words and negations are considered. Her results, using the rule-based approach achieved 69.6% accuracy for positive sentences and 71.0% for negative sen-
tences. She attributes this to the observation that “negative opinions ( . . . ) are more transparent than the positive opinions”.

Wiegand et al. (2010) took a closer look at the issue of negation in sentiment analysis. The authors observed that bag-of-words models, which do not explicitly model negation and which lack linguistic analysis perform reasonably well, especially in document-level analysis. Although, they also point to work using a parser to determine the scope of negation, but do not report on the final results of this. The authors observed that negation words are more important for the classification than diminishers. They also point to words containing negations, which can only be determined using a morphological analysis. But they cite only few examples where this has been carried out and the impact on the polarity classification is unclear. While their analysis primarily focuses on English, they point to language-specific phenomena, which makes the treatment of negations more difficult in languages such as German, where the negation can occur before or after the word(s) it refers to.

3 Experimental Setup

The major functionality of our processing pipeline (see Figure 1) is the computation of sentiment for comments related to "Deutsche Bahn" and "travelling" based on the Germeval 2017 dataset (Wojatzki et al., 2017). We determine the sentiment as "negative", "neutral" and "positive". In order to classify the comments, we split them into sentences and chunks first. Then, each chunk gets lemmatized. For measuring the sentiment of each chunk, we take both the term-frequency as well as the polarity of each word into account. If a negation occurs before a sentiment-bearing word, we reverse the sentiment of the respective word. Further, we use the SentiWS lexicon (Remus et al., 2010). The latter includes polarities in the interval of [-1,1]. We manually created a list of negations, as well as a list of synonyms for "Deutsche Bahn" and "travelling", which was based on the OpenThesaurus. The resources we manually created are provided to the community.

5 Measuring Polarity

We determine the polarity of each document in a three-step pipeline. During preprocessing, we split the comments into individual sentences and chunks. We analyse them individually and combine this analysis in order to measure the sentiment of each sentence. The sentiment of the whole document is then based on the accumulated sentiment of individual words and sentences.

5.1 Preprocessing

Each comment is split into sentences, chunks and tokens. For the sentence-tokenization we use the Maxent annotator, for lemmatization and chunking we use the TreeTagger.

5.2 Measuring Sentiment

We compute a term-document matrix (tdm) to determine the term-frequencies of each word in the chunks. Then, we match the words with the entries in the sentiment dictionary. For each chunk we extract the single polarities with regard to applicable word-frequencies by multiplying the two values. Negations reverse the polarity of the following sentiment-bearing words in each chunk. We combine the polarities of each chunk, sentence and comment to determine the final sentiment score.

Applicable sentiment scores of chunks are used as polarities to compose \( \sigma_i \), that represents the sentiment score of sentence \( i \). We combine polarities in \( \sigma_i \) by taking the arithmetic mean of the polarities (\( \bar{x}_i \)) multiplied by a weighting factor. This factor corresponds to half of the percentage of applicable chunks in which sentiment can be detected (\( \rho_i \)). We generate a random variable \( Y \) that follows the uniform distribution with parameters zero and the maximal amount of chunks that are combined per sentence (\( \max(y_i) \)). \( P(Y \leq y_i) \) represents the probability that \( Y \) takes on a value less or equal to \( y_i \).
Therefore, the weighting factor corresponds to half of the probability that $Y$ takes on a value less or equal to the amount of chunks that are combined in sentence $i$ ($y_i$). Then, we locate topic related sentiment in each sentence through the dictionary of topic words and their synonyms. If we are not able to locate a word from the dictionary in sentence $i$, we set the respective polarity to zero. The final sentiment score of the whole comment is obtained in a similar fashion, by applying the polarity calculation to sentences.

$$\sigma_i = \tau_i \cdot \left( \frac{1}{2} \rho_i + \frac{1}{2} \cdot P(Y \leq y_i) \right), \quad Y \sim U(0, \max(y_i))$$

(1)

5.3 Classification

For the final classification for each comment into “negative”, “neutral” and “positive” we use the results of Formula 1. Intuitively, values above 0 would result in positive comments and values below 0 would be classified as negative values. Nevertheless, we experimented with various thresholds for $\sigma_i$ and experimentally determined the best set of thresholds for “negative”, “positive” and “neutral” using the development set. These are also used for the test data. Scores for the best set of thresholds on the development set are presented in Table 1 below. As can be seen, the intuitive thresholds achieve worse results than thresholds which are slightly larger and smaller than 0.

6 Results

In the following, we present results on the development set and discuss some of the observed errors.

6.1 Evaluation

Table 1 shows results from the provided development data with various thresholds (th) and uses indicators micro and macro F1 for evaluation. The first line shows results in which comments that have a polarity of zero are classified as neutral. F1 measures for groups positive and negative are the best in line one. An overall high F1 indicator is shown in line two. But F1 measures for groups positive and negative are lower than 0.2 in this line. It derives from an unbalanced size of groups.

<table>
<thead>
<tr>
<th>th pos</th>
<th>th neg</th>
<th>micro F1</th>
<th>macro F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>dev</td>
<td>0.01</td>
<td>0.544</td>
<td>0.544</td>
</tr>
<tr>
<td></td>
<td>0.001</td>
<td>0.666</td>
<td>0.666</td>
</tr>
</tbody>
</table>

Table 1: Comparison between thresholds

6.2 Error Analysis

Table 2 shows the confusion matrix for the development set. When looking at the wrongly classified instances, we observe several problems that would need further work. One issue lies with the resources we employed in combination with the data and the domain. For example, if "Bahn" (railway) is written in lower case letters ("bahn") the TreeTagger sets the verb "bahnen" as lemma instead of the noun "Bahn". These instances are not regarded as related to travelling by train. The word "Streik" (strike) is marked as minimally negative in the SentiWS dictionary. Considering the context of Deutsche Bahn or travelling, the polarity of "Streik" should be far more negative. The word "Störung" (engl. malfunction) is not included in SentiWS. But it is sentiment-bearing in this context. The phrase "Streik beenden" (ending a strike) has negative polarities in SentiWS. But connecting these words to "Streik beenden" gives them a positive polarity. This is not recognized by the predicted vs. actual

<table>
<thead>
<tr>
<th></th>
<th>positive</th>
<th>negative</th>
<th>neutral</th>
</tr>
</thead>
<tbody>
<tr>
<td>positive</td>
<td>32</td>
<td>49</td>
<td>145</td>
</tr>
<tr>
<td>negative</td>
<td>33</td>
<td>114</td>
<td>137</td>
</tr>
<tr>
<td>neutral</td>
<td>84</td>
<td>426</td>
<td>1355</td>
</tr>
</tbody>
</table>

Table 2: Confusion Matrix on the dev set using the best threshold.

small
algorithm. Also, the topic dictionary is not comprehensive enough. Hashtags that are associated with "deutsche Bahn" or travelling (e.g. #Bahn) are not added to the dictionary, as well as Twitter accounts like @DB_Bahn or @Regio_NRW, which are also missing.

When looking at the confusion matrix, we observe that both positive and negative texts have been often confused with neutral posts by our algorithm. A closer look at the instances reveals, that some of them are indeed neutral. One example is: Bahnhof in Wittenberg: Info-Umzug zum Streikende — Wittenberg/Gräfenhainichen - Mitteldeutsche Zeitung (BILD: Baumbach) Alexander Baumbach Am Wittenberger Bahnhof gibt es jetzt Reiseauskünfte nur noch im Container. Bis zum Bezug des neuen Bahnhofgebäudes im nächsten Jahr wird dies auch so bleiben. Wittenberg Pünktlich zum Ende des Lokführer-Streik. This has been marked as neutral by our algorithm, but the gold standard says it is positive, probably due to the end of the strike. But we tend to agree, that the post itself is primarily neutral.

Another instance such as RT @nhitastic: Wisst ihr was geil ist? WLAN in der deutschen Bahn! @DB_Bahn haha is also marked as neutral, whereas it is supposed to be positive. Looking at the details, it is obvious, that "sounds" such as haha could point either to a positive sentiment or could be regarded as ironic, which would make this post negative rather than positive.

An example for a negative post, that was classified as neutral by our algorithm is: Bericht über hunderte Funklöcher bei der Bahn https://t.co/cMoUIaSHOk

Instances where our algorithm classified a post as negative, whereas it was positive can be attributed to our lexicon-based approach. For example in GDL-Streik beendet: Warum Bahnkunden dem Frieden nicht trauen Pfingsten mit der Bahn kann kommen und Millionen Bahnkunden atmen auf. Die Lokführer haben ihren Streik abgeblasen. Reisende bleiben trotzdem skeptisch

phrases such as “nicht trauen” (not trusting) and words such as “skeptisch” (sceptic) tend to be more negative than positive. Even though this post is on the strike, customers are still wary, which is the major part of this post and therefore, this post is classified as negative, whereas the gold standard marks it as positive.

The neutral class was reliably classified. There were similar amounts of instances misclassified as positive or negative.

One example being: Der Schweizer Bahn-Vierer musste sich beim Weltcup im kolumbianischen Cali nur den Russen geschlagen geben. https://t.co/Q5VUos57w #srfra

Which actually has no relation to travelling by train and therefore is not relevant for the task at hand. Another example is: Veränderung bei Pendlern in London: Tiefstehende Sonne sorgt bei Bahn für Verspätungen https://t.co/0PgTR06YS2

While “sun” normally would be related to a positive sentiment, in this case, as it caused delays it would be negative. In combination our lexicon-based approach marked it as neutral.

7 Conclusion and Future Work

In this paper, we presented our contribution to the GermEval 2017 document level polarity task using R. Our pipeline is based on various lexical resources, which determine positive and negative words. Additionally, we consider negations in order to switch the sentiment of the respective phrase. We also used various linguistic preprocessing methods such as a chunker to allow for a better treatment of the scope of the negations. Lemmatization reduced the search space for sentiment-bearing words.

We aim to continuously improve this pipeline. A major improvement for the future would be to create a domain-dependent sentiment lexicon in order to capture specific words and phrases which in this particular context have a stronger polarity than in others. Also, instead of just switching the polarity of a word based on the existence of negation words, a more fine-grained approach would be meaningful. Taking intensifiers into account would also be beneficial. Additionally, the linguistic pre-processing tools we used were not adapted for social media text. Finally, adapting the rule-based method to using machine-learning methods would greatly improve the performance of our pipeline.

Acknowledgements

We would like to thank Magdalena Bergold, Cheuk-Hei Chin, Martin Czernin, Viktoria Gaus, Benjamin Lossau, David Michalski, Yeliz Ovic, Björn Severitt, Daniela Di Schiena and Nikolai Spuck who attended Christoph Becker’s course "Textmining with R" and created the very first version on which these results are based.
References


HU-HHU at GermEval-2017 Sub-task B:
Lexicon-Based Deep Learning for Contextual Sentiment Analysis

Behzad Naderalvojoud
DFG SFB 991
Hacettepe University
Ankara, Turkey
n.behzad@hacettepe.edu.tr

Behrang Qasemizadeh
DFG SFB 991
Universität Düsseldorf
Düsseldorf, Germany
zadeh@phil.hhu.de

Laura Kallmeyer
DFG SFB 991
Universität Düsseldorf
Düsseldorf, Germany
kallmeyer@phil.hhu.de

Abstract

This paper describes the HU-HHU system that participated in Sub-task B of GermEval 2017, Document-level Polarity. The system uses 3 kinds of German sentiment lexicons that are built using translations of English lexicons, and it employs them in a neural-network based context-dependent sentiment classification method. This method uses a deep recurrent neural network to learn context-dependent sentiment weights to change the lexicon polarity of terms depending on the context of their usage. The performance of the system is evaluated using the benchmarks provided by the task’s organizers.

1 Introduction

Sentiment lexicons are known as useful language resources in sentiment analysis systems that determine the sentimental orientation of terms out of context. As manually creating such lexicons is expensive and time-consuming, one possible solution is to translate English resources into other languages (Waltinger, 2010; Ucan et al., 2016). However, natural language is ambiguous and word by word translation cannot achieve satisfying results; terms can possess more than one sentiment and can express different sentiments with respect to the context. In this case, sense-based sentiment lexicons, like SentiWordNet (SWN) (Esuli and Sebastiani, 2006), assign polarities to word-senses instead of words. However, the number of word-senses cannot necessarily be the same for two terms that are translations of each other. Moreover, not all word-senses of a term may express subjectivity in a language (Akkaya et al., 2009). Therefore, we generated a sentiment lexicon for German that takes into account SWN synsets. To map subjective synsets to German terms, we extended the approach used in the HUMIR1 project (Naderalvojoud et al., 2017) for applying it to German language. This approach creates a cross-lingual sense mapping between the SWN synsets and German terms and produces a single polarity value for each term. This value indicates the strength of subjectivity according to the number of mapped synsets. In fact, the polarity and the number of German synsets associated with each term constitute the domain of German terms’ sentiments.

Besides this lexicon, we also employ two other German sentiment lexicons proposed in (Waltinger, 2010) that are translations of English Subjectivity Clues (Wilson et al., 2005) and SentiSpin (Takamura et al., 2005) lexicons. Three online English-to-German translation softwares have been used in constructing the German lexicons; a German term that appears in most of translation results is selected as a translation of the given English term. While the polarity values of German Subjectivity clues are assigned manually, they are inherited from the corresponding English resource for German SentiSpin.

The sentiment lexicons generated is employed in a context-dependent sentiment analysis system that uses a deep recurrent neural network (RNN) to capture the contextual sentiment modifications that can change the prior known sentiments of terms with respect to the context. After describing the construction of the proposed German SentiWordNet lexicon in Section 2, we explain the sentiment analysis system we used in Section 3. Section 4 reports the evaluation results. Finally, Section 5 presents the conclusion.

2 German SentiWordNet Lexicon

The proposed German SentiWordNet lexicon is constructed by the following three main steps using the Open Subtitle Corpus (Tiedemann, 2009) for

1http://humir.cs.hacettepe.edu.tr/projects/lsa.html
the German–English language pair: (1) We first generate a cross-lingual distributional/statistical model to represent subjective English terms in the German language vector space. In this model, each English term is represented by a distributional semantic vector whose elements are German terms (Naderalvoojoud et al., 2017). (2) The generated model is used to represent the SWN synsets. To represent synsets, the semantic vectors of the synset terms are summed up. (3) Synset mapping is applied to the reduced semantic vectors for mapping German terms to subjective SWN synsets. We suppose that German terms with high frequency in the semantic vector are more likely associated with the given synset.

As a result of this approach, we achieved two lexicons of 14,309 (named SWN1) and 43,790 (named SWN2) German subjective terms by using two different corpora having 70,534 and 13,883,398 movie subtitles, respectively.

3 Context-Dependent Sentiment Analysis Using Deep Learning

We use deep learning to capture the implicit sentiment knowledge contained in the semantic/syntactic structure of a sentence. In fact, the prior sentiment of terms in the lexicon can be changed based on the negation, intensification, or semantic structure of terms in the context. For example, the positive sentiment of “good” is shifted to negative in the sentence “Nobody gives a good performance in the team” by the word “nobody”. A similar situation can be seen for the word “great” in the sentence “He was a great liar” and its positive sentiment (based on the lexicon) is shifted to negative. Thus, the use of sentiment lexicons without consideration of the context cannot achieve a satisfying result.

In the sentiment analysis method we use, the contextual sentiment knowledge is combined with the terms’ prior sentiments. To this end, we employ a context-sensitive lexicon-based method proposed in (Teng et al., 2016). In this approach, the sentiment score of a sentence is computed based on the weighted sum of the polarity values of the subjective terms obtained from the lexicon. The learned weights are considered as context-dependent features that modify the prior polarity values of terms with respect to the context. Therefore, the effects of negation, shifting and intensification can be considered in the sentiment classification task. The overall structure of the model is simply shown in Eq. 1.

\[
\text{Score} = \sum_{k=1}^{N} \gamma_k \times \text{LexScore}(t_k) + b \tag{1}
\]

In Eq. 1, \(N\) denotes the number of subjective terms in the sentence, \(\text{LexScore}(t_k)\) is the polarity value of term \(t_k\) in the lexicon, \(\gamma_k\) is the context-dependent weight of term \(t_k\) and \(b\) is the sentence bias score.

To Learn \(\gamma\) and \(b\), following (Teng et al., 2016), we employ a recurrent neural network (RNN) model with bidirectional long-short-term-memory (BiLSTM) cells (Graves et al., 2013; Sak et al., 2014) for extracting the semantic composition features.

We use the sentiment annotated data provided by the task organizers (Wojatzi et al., 2017) for training the model. The German FastText pre-trained word embeddings are used in generating the model in combination with the German sentiment lexicons. We tune hyper-parameters of our model using the obtained classification results on the development set.

4 Evaluation Report

We evaluate the proposed sentiment model using customer reviews about “Deutsche Bahn” provided by the task organizers. Table 1 shows the statistics of data in three categories, “positive”, “negative”, and “neutral”. In order to show the significance of contextual sentiment analysis, we also indicate the contextual ambiguity of the training set by relying on the occurrence of subjective terms in irrelevant reviews. For example, in Table 1, while column “NegInPos” shows the percentage of positive reviews with negative clue terms, the column “PosInNeg” indicates the occurrence of positive clue terms in negative reviews. The most important point is the occurrence of subjective terms in the neutral reviews (column “SubInNeu”) which can make it hard to distinguish neutral reviews from subjective ones. The other observation is that the number of neutral reviews outnumbers the positive and negative...
tive ones. For example, only 6% of train reviews are positive, whereas 68% are neutral.

Table 1: Statistics of dataset

<table>
<thead>
<tr>
<th>Data Split</th>
<th>All</th>
<th>Pos 6%</th>
<th>Neg 26%</th>
<th>Neu 68%</th>
</tr>
</thead>
<tbody>
<tr>
<td>train</td>
<td>19432</td>
<td>1179</td>
<td>5045</td>
<td>13208</td>
</tr>
<tr>
<td>dev</td>
<td>2369</td>
<td>148</td>
<td>589</td>
<td>1632</td>
</tr>
<tr>
<td>test-syn</td>
<td>2566</td>
<td>105</td>
<td>780</td>
<td>1681</td>
</tr>
<tr>
<td>test-dia</td>
<td>1842</td>
<td>108</td>
<td>497</td>
<td>1237</td>
</tr>
<tr>
<td>sum</td>
<td>26209</td>
<td>1540</td>
<td>6911</td>
<td>17758</td>
</tr>
</tbody>
</table>

As the train set is imbalanced, the performance of the classification model can tend towards the majority class (Naderalvojoud et al., 2015). As a result of this, the micro F-measure value (which is the shared task evaluation metric) is affected by the majority class. Hence, we use the macro F-measure value along with the micro score in the evaluation.

Table 2 indicates the best results on the development set achieved from the shared task baseline system (SVM) and our context-dependent system (RNN). We select the model that produces the best results on the development set for applying to test set. As two lexicons generated from SWN achieve the best results in comparison to two other lexicons, we constructed two models according to these lexicons for testing.

From the results shown in Table 2, the RNN model outperforms the baseline system in all three classes. While the baseline system yields weak F-measure values for positive and negative classes, the RNN-based system achieves F-measure values of 0.4533 and 0.6254. Despite the lack of positive instances in the train set (6%), the RNN model can achieve much better results than SVM in combination with the proposed German SWN lexicons. This can be also observed in the negative class in which the F-measure value increases from 0.2212 in SVM to 0.6254 in RNN. Overall, the change in Positive–Negative macro F-measure value from 0.1173 (in Baseline-SVM) to 0.5394 (in SWN1-RNN) clearly shows the effect of the proposed lexicon-based context-dependent sentiment analysis method. It is worth noting that a German lexicon has been also used in the baseline system, however, this system has not made an impact on the sentiment classification result as much as the proposed RNN model has made. Furthermore, while the German SentiSpin lexicon does not improve the performance of the RNN model in the positive class, the proposed German SWN lexicons significantly improve its performance. Although the German subjectivity clue lexicon performs better than SentiSpin, the proposed German SWN1 and SWN2 outperform it by 7% and 4% in MacroF1(PN), respectively.

As the neutral class is the majority class in the train set, all systems yield high F-measure values for this class. Nevertheless, the RNN model in combination with two German SWN lexicons achieves the best results in terms of macro F-measure value (0.6452, SWN1-RNN) and micro F-measure value (0.7873, SWN2-RNN) over all classes.

Tables 3 and 4 show the results of the baseline and proposed systems on the synchronic and diachronic test sets, respectively. From these results, we can observe that the proposed system outperforms the baseline method by using all German sentiment lexicons. In synchronic test set, while the SWN1-RNN achieves the best macro F-measure value (0.4907), SWN2-RNN yields the best micro F-measure value (0.7494). In diachronic test set, the RNN model with both German SWN lexicons achieves the best micro F-measure values. However, they do not maintain this superiority and the RNN model with German Subjectivity clue lexicon gives the best macro F-measure value (0.5211). This may arise from the fact that the polarity values of German Subjectivity Clues are manually assigned. As a result, the proposed context-dependent sentiment analysis system performs well in combination with the German SWN lexicons and remarkably outperforms the baseline SVM model.

5 Conclusion

This paper presented the sentiment analysis approach of HU-HHU system in the GermEval 2017 shared task. In this approach, an RNN model is used to learn the context-dependent sentiment weights that can change the lexicon polarity of terms depending on the context. As shown in the empirical evaluations, compared to the baseline system, this approach significantly improves the performance of the sentiment classification task.

References

### Table 2: The best results on the development set

<table>
<thead>
<tr>
<th>Lexicon-Model</th>
<th>F1pos</th>
<th>F1neg</th>
<th>F1neu</th>
<th>MacroF1(PN)</th>
<th>MacroF1(all)</th>
<th>MicroF1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline-SVM</td>
<td>0.0134</td>
<td>0.2212</td>
<td>0.8244</td>
<td>0.1173</td>
<td>0.3530</td>
<td>0.7092</td>
</tr>
<tr>
<td>SWN1-RNN</td>
<td><strong>0.4533</strong></td>
<td><strong>0.6254</strong></td>
<td><strong>0.8568</strong></td>
<td><strong>0.5394</strong></td>
<td><strong>0.6452</strong></td>
<td>0.7847</td>
</tr>
<tr>
<td>SWN2-RNN</td>
<td>0.4381</td>
<td>0.6086</td>
<td>0.8602</td>
<td>0.5234</td>
<td>0.6356</td>
<td>0.7873</td>
</tr>
<tr>
<td>SentiSpin-RNN</td>
<td>0.0127</td>
<td>0.6075</td>
<td>0.8525</td>
<td>0.3101</td>
<td>0.4909</td>
<td>0.7716</td>
</tr>
<tr>
<td>SubjectivityClues-RNN</td>
<td>0.4112</td>
<td>0.5932</td>
<td>0.8591</td>
<td>0.5022</td>
<td>0.6212</td>
<td>0.7838</td>
</tr>
</tbody>
</table>

### Table 3: Results on synchronic test set

<table>
<thead>
<tr>
<th>Lexicon-Model</th>
<th>MacroF1</th>
<th>MicroF1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline-SVM</td>
<td>0.3325</td>
<td>0.6730</td>
</tr>
<tr>
<td>SWN1-RNN</td>
<td><strong>0.4907</strong></td>
<td>0.7366</td>
</tr>
<tr>
<td>SWN2-RNN</td>
<td>0.4806</td>
<td><strong>0.7494</strong></td>
</tr>
<tr>
<td>SentiSpin-RNN</td>
<td>0.4764</td>
<td>0.7159</td>
</tr>
<tr>
<td>SubjectivityClues-RNN</td>
<td>0.4718</td>
<td>0.7357</td>
</tr>
</tbody>
</table>

### Table 4: Results on diachronic test set

<table>
<thead>
<tr>
<th>Lexicon-Model</th>
<th>MacroF1</th>
<th>MicroF1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline-SVM</td>
<td>0.3539</td>
<td>0.6894</td>
</tr>
<tr>
<td>SWN1-RNN</td>
<td>0.5165</td>
<td><strong>0.7362</strong></td>
</tr>
<tr>
<td>SWN2-RNN</td>
<td>0.5036</td>
<td><strong>0.7362</strong></td>
</tr>
<tr>
<td>SentiSpin-RNN</td>
<td>0.4482</td>
<td>0.7176</td>
</tr>
<tr>
<td>SubjectivityClues-RNN</td>
<td><strong>0.5211</strong></td>
<td>0.7323</td>
</tr>
</tbody>
</table>

---

Volume 1, pages 190–199. Association for Computational Linguistics.


---

21
Abstract
This paper describes our submissions to the GermEval 2017 Shared Task, which focused on the analysis of customer feedback about the Deutsche Bahn AG. We used sentence embeddings and an ensemble of classifiers for two sub-tasks as well as state-of-the-art sequence taggers for two other sub-tasks. Relevant aspects to reproduce our experiments are available from https://github.com/UKPLab/germeval2017-sentiment-detection.

1 Introduction
For many companies, customer feedback is an important source for identifying problems affecting their services. Although customer feedback may be obtained by interviewing single customers or conducting larger studies using questionnaires, those are often cost-intensive. Instead, it is much cheaper to crawl customer feedback from the web, for example from social media platforms like Twitter, Facebook, or even news pages. In contrast to interviews or questionnaires, crawled data is often noisy and does not necessarily cover specific company-related topics. Due to the vast amount of available data on the web, it is crucial to analyze relevant documents and extract the feedback automatically.

The GermEval 2017 Shared Task (Wojatzki et al., 2017) focuses on the automated analysis of customer feedback about the Deutsche Bahn AG (DB) in four subtasks, namely (A) relevance classification of documents, (B) identification of the document-level polarity, (C) identification of certain aspects in a single document as well as predicting their category and polarity, and (D) extraction of the exact phrase of a single aspect.

For example, the tweet

@RMVdialog hey, wann fährt denn nach der Störung jetzt die nächste Bahn von Glauberg nach Ffm?

has the following gold-standard annotations for Tasks A-D, respectively:

(A) true
(B) neutral
(C) Sonstige Unregelmäßigkeiten – negative
(D) Störung

We participated in all subtasks of the shared task. For Tasks A, B, and C we trained models on document-level representations using a classifier ensemble. As Task D can be modeled as a sequence tagging task, we used a state-of-the-art deep neural network tagger with a conditional random field at the output layer.

This work is structured as follows. Section 2 gives an overview of the data. Section 3 details the two main modeling approaches we made use of. Section 4 describes our experimental set-ups, and presents and discusses our results for a selection of well-performing models. We conclude in Section 5.

2 Data
The data provided for this shared task contains ≈ 22,000 German messages from various social media and web sources and has been annotated in a joint project between Technische Universität Darmstadt and DB. In addition to the provided data we used several external resources for training word and sentence embeddings and computing task specific features.

2.1 Task Specific Data
The shared task data contains annotations about the relevance R of a message (Task A) and its sentiment polarity (B), either positive P, negative NG, or neutral NT. Relevant messages contain further annotations about their aspects, with the aspect category and sentiment polarity (C) and its exact...
Table 1: Class distributions for task A and B in %

<table>
<thead>
<tr>
<th>Category</th>
<th>#</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allgemein</td>
<td>13892</td>
</tr>
<tr>
<td>Zugfahrt</td>
<td>2421</td>
</tr>
<tr>
<td>Sonstige Unregelmäßigkeiten</td>
<td>2112</td>
</tr>
<tr>
<td>Atmosphäre</td>
<td>1576</td>
</tr>
<tr>
<td>Sicherheit</td>
<td>962</td>
</tr>
<tr>
<td>Ticketkauf</td>
<td>741</td>
</tr>
<tr>
<td>Service und Kundenbetreuung</td>
<td>551</td>
</tr>
<tr>
<td>Connectivity</td>
<td>390</td>
</tr>
<tr>
<td>Informationen</td>
<td>388</td>
</tr>
<tr>
<td>Auslastung und Platzangebot</td>
<td>304</td>
</tr>
<tr>
<td>DB App und Website</td>
<td>252</td>
</tr>
<tr>
<td>Komfort und Ausstattung</td>
<td>166</td>
</tr>
<tr>
<td>Barrierefreiheit</td>
<td>89</td>
</tr>
<tr>
<td>Toiletten</td>
<td>54</td>
</tr>
<tr>
<td>Image</td>
<td>54</td>
</tr>
<tr>
<td>Gastronomisches Angebot</td>
<td>47</td>
</tr>
<tr>
<td>Reisen mit Kindern</td>
<td>46</td>
</tr>
<tr>
<td>Design</td>
<td>37</td>
</tr>
<tr>
<td>Gepäck</td>
<td>15</td>
</tr>
<tr>
<td>QR-Code</td>
<td>1</td>
</tr>
<tr>
<td>Total</td>
<td>24098</td>
</tr>
</tbody>
</table>

Table 2: The number of aspects per category.

For Tasks C and D, the data contains 24,098 aspects in total, which are classified into 20 different categories. Table 2 shows the number of aspects for each category. We observe that the data is highly skewed here, with more than 57% of all aspects being of category “Allgemein”. Table 3 shows the distribution of positive, negative, and neutral aspects in the data.

Furthermore, not every aspect can be matched to an exact phrase (target). In 44% of the cases, a category and the polarity is assigned to a message without having a target. For these cases, the target is annotated by NULL.

### 2.2 External Sources

We use several external data sources for training various word and sentence embeddings, namely a German Wikipedia corpus (Al-Rfou et al., 2013) and a German Twitter corpus (Cieliebak et al., 2017). The Wikipedia corpus is publicly available and contains already tokenized data. We use a crawler published along with the Twitter corpus, to obtain the actual texts of the tweets. This results in a corpus containing 7464 tweets, which we then tokenized using the Tweet Tokenizer from NLTK (Bird et al., 2009).

We also made use of an English Twitter sentiment corpus of around 40K tweets (Rosenthal et al., 2017), each annotated with positive, negative, or neutral stance, just as the German data. Our hope was that this would provide a strong additional signal from which our learners could induce the sentiment of a tweet, be it English or German. To make use of this additional data, we projected our word and sentence embeddings (see below) in a bilingual German-English embedding space so that they are comparable.1 We used CCA (Faruqui and Dyer, 2014) for this, which requires independently constructed language specific embeddings and word translation pairs (such as (Katze, cat)) to allow projecting vectors into a joint space. The word translation pairs were induced from the Europarl corpus (Koehn, 2005).

### 3 Methods

In what follows, we describe, on a general level, our approaches to Tasks A and B (Section 3.1) and Task D (Section 3.2), respectively. For Task C, we mixed between the approaches outlined in Sections 3.1 and 3.2 in our experiments. We relegate the corresponding model description to Section 4.

#### 3.1 Sentence Embeddings and Classifier Ensemble

We used a unified and minimally expensive (in terms of feature engineering) approach to tackle Tasks A and B, which both concern the classification of documents into categories. We tokenized...
each document and converted it to an embedding via the tools Sent2Vec (Pagliardini et al., 2017) and SIF (Arora et al., 2017). Both of these tools aspire to improve upon the simple average word embedding baseline for sentence embeddings, but are conceptually simple. We trained Sent2Vec on the union of German Wikipedia data as well as a Twitter corpus and the task specific data of the Shared Task. For SIF, we first created word embeddings with the standard skip-gram model of Word2Vec (Mikolov et al., 2013), and then generated sentence embeddings from these via specific SIF parametrizations outlined below. We train Word2Vec on the same data sources as Sent2Vec.

After converting documents to embeddings of particular sizes $d$, we train a classifier that maps representations in $\mathbb{R}^d$ to one of $N$ classes, where $N = 2$ for Task A and $N = 3$ for Task B. We use the stacked learner from Eger et al. (2017) as a classifier. This is an ensemble based system that uses several base classifiers from scikit-learn and a multilayer perceptron as a meta-classifier to combine the predictions of the base classifiers.

### 3.2 (MTL) Sequence Tagging

Task D is naturally modeled as sequence tagging task, that is, it can be framed as the problem of tagging each element in a sequence of tokens $x_1, \ldots, x_T$ with a label $y_1, \ldots, y_T$. We used the most recent state-of-the-art sequence tagging frameworks (Lample et al., 2016; Ma and Hovy, 2016), which consist of a neural network (bidirectional) LSTM tagger that uses word and character level information as well as a CRF layer on top that accounts for dependencies between successive output predictions. Moreover, since multi-task learning (MTL) settings in which several tasks are learned jointly have been reported to sometimes outperform single-task learning (STL) scenarios, we directly allow for inclusion of several tasks during training and prediction time. Our approach builds here upon the architecture of Søgaard and Goldberg (2016) in which different tasks feed from particular levels of hidden layers in a deep LSTM tagger. We used the framework (Kahse, 2017) that accounts for dependencies between successive output predictions. Moreover, since multi-task learning (MTL) settings in which several tasks are learned jointly have been reported to sometimes outperform single-task learning (STL) scenarios, we directly allow for inclusion of several tasks during training and prediction time. Our approach builds here upon the architecture of Søgaard and Goldberg (2016) in which different tasks feed from particular levels of hidden layers in a deep LSTM tagger. Our employed framework (Kahse, 2017) extends Søgaard and Goldberg (2016) in that we include both character and word-level information as well as implement CRF layers for each task, as mentioned already. Note that we could in principle train all four Shared Task tasks in a single architecture, possibly with Tasks A and B feeding from lower layers of the deep LSTM, because the tasks satisfy some of the requirements that have often been attributed to successful MTL, such as relatedness of tasks and natural task hierarchy.

To illustrate, for Task D, the goal is to extract the relevant phrase to be classified in Task C. We frame this as a token-level BIO tagging problem in which each token is labeled with one of three classes from $\{I, O, B\}$. That is,

$$\text{Notrufsystem : 250 Funkl"ocher bei } \ldots$$

B I I I O ... retrieves the target phrase Notrufsystem : 250 Funkl"ocher from the document.

### 4 Experiments

#### 4.1 Tasks A and B

**Approach:** We train models with document-level features using the stacked learner. We focus on the comparison of different word and sentence embeddings, and additional polarity features computed using a lexical resource described in Waltinger (2010) for Task B. For document embeddings, we evaluate average word vectors, besides the approaches mentioned above. Furthermore, we ran experiments with combinations of different word and sentence embeddings. For these, we compute average word vectors for a single document and concatenate it with the respective sentence embedding.

**Hyperparameters:** We compare Word2Vec and 100 dimensional Komninos word embeddings (Komninos and Manandhar, 2016), and 500 dimensional Sent2Vec and SIF sentence embeddings as described before.\(^2\) Word2vec skip-gram embeddings are computed for dimensions $d = 50, 100, 500$. In addition, we compare two SIF embeddings computed with different input word embeddings. One was computed from the German data directly and another one by projecting the

\(^2\)An updated data set was released on August, 10th. For our submissions, we retrained all models on the new data.

\(^3\)For computing the SIF embeddings, we use the word weighting parameter $a = 0.01$ and, we subtract the first $r = 2$ principal components. See the original paper for details.
Table 4: Task A results

<table>
<thead>
<tr>
<th>Model</th>
<th>Micro F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>0.882</td>
</tr>
<tr>
<td>W2V (d = 50)</td>
<td>0.883</td>
</tr>
<tr>
<td>W2V (d = 500)</td>
<td>0.897</td>
</tr>
<tr>
<td>S2V</td>
<td>0.885</td>
</tr>
<tr>
<td>S2V + W2V (d = 50)</td>
<td>0.891</td>
</tr>
<tr>
<td>S2V + K + W2V(d = 50)</td>
<td>0.890</td>
</tr>
<tr>
<td>SIF (DE)</td>
<td>0.895</td>
</tr>
<tr>
<td>SIF (DE-EN)</td>
<td>0.892</td>
</tr>
</tbody>
</table>

Table 5: Task B results

<table>
<thead>
<tr>
<th>Model</th>
<th>Micro F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>0.709</td>
</tr>
<tr>
<td>W2V (d = 50)</td>
<td>0.736</td>
</tr>
<tr>
<td>W2V (d = 500)</td>
<td>0.753</td>
</tr>
<tr>
<td>S2V</td>
<td>0.748</td>
</tr>
<tr>
<td>S2V + W2V (d = 50)</td>
<td>0.744</td>
</tr>
<tr>
<td>S2V + K + W2V(d = 50)</td>
<td>0.749</td>
</tr>
<tr>
<td>SIF (DE)</td>
<td>0.759</td>
</tr>
<tr>
<td>SIF (DE-EN)</td>
<td>0.765</td>
</tr>
</tbody>
</table>

Table 6: Task B results with polarity features

<table>
<thead>
<tr>
<th>Model</th>
<th>Macro F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>0.478</td>
</tr>
<tr>
<td>MTL-Adam (d = 50)</td>
<td>0.438</td>
</tr>
<tr>
<td>STL-Adam (d = 50)</td>
<td>0.458</td>
</tr>
<tr>
<td>STL-Adam (d = 100)</td>
<td>0.488</td>
</tr>
<tr>
<td>STL-Adam (d = 100) + POS-Tags</td>
<td>0.494</td>
</tr>
<tr>
<td>STL-AdaDelta (d = 100)</td>
<td>0.543</td>
</tr>
<tr>
<td>STL-AdaDelta (d = 100) + POS-Tags</td>
<td>0.554</td>
</tr>
</tbody>
</table>

Table 7: Task D results

<table>
<thead>
<tr>
<th>Model</th>
<th>Micro F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>0.709</td>
</tr>
<tr>
<td>W2V (d = 50)</td>
<td>0.748</td>
</tr>
<tr>
<td>W2V (d = 500)</td>
<td>0.756</td>
</tr>
<tr>
<td>S2V</td>
<td>0.748</td>
</tr>
<tr>
<td>S2V + W2V (d = 50)</td>
<td>0.755</td>
</tr>
<tr>
<td>S2V + K + W2V(d = 50)</td>
<td>0.751</td>
</tr>
<tr>
<td>SIF (DE)</td>
<td>0.748</td>
</tr>
<tr>
<td>SIF (DE-EN)</td>
<td>0.757</td>
</tr>
</tbody>
</table>

German data into a shared embedding space with English embeddings as described before.

**Results:** The results of the models better than the baseline are reported in Tables 4 and 5. As can be seen, all models only slightly outperform the baseline in Task A. For Task B, all models trained on the stacked learner beat the baseline substantially even when using only plain averaged word embeddings. We furthermore trained models on additional polarity features for Task B as mentioned before. For this, we look up all positive, negative, and neutral words in a document and compute a three-dimensional polarity vector by using the total count of found words. These are concatenated to the respective document representation. Adding the polarity features improved the results for all models except for those using SIF embeddings (Table 6).

**Discussion:** Unexpectedly, the model using the averaged Word2Vec embedding performs best for Task A, even though the other embeddings created by Sent2Vec or SIF have the same dimension (500). A reason for this may be chance or the Twitter data. As the experiments of Pagliardini et al. (2017) confirm, averaged Word2Vec embeddings perform rather well for a similarity task on Twitter data. However, we observe that particularly SIF outperforms average word embeddings for Task B. We also observe that the joint EN-DE embeddings improve results for Task B (+0.6% and +0.9%, respectively) but lead to a drop in performance for Task A (-0.3%). This is in line with the common observation that the bilingual signal may provide an additional source of both useful and noisy, irrelevant, or even hurtful information (Faruqui and Dyer, 2014; Eger et al., 2016).

4.2 Task D

**Approach:** We tackle this task with our sequence tagging framework and evaluate on the dev set using the macro F1 score.

**Hyperparameters:** We use Word2Vec embeddings of \(d = 50, 100\) trained on German Wikipedia, Twitter, and the shared task data. We also incorporate 20 dimensional skip-gram embeddings for POS-tags, trained on the data of the shared task and concatenate them with the corresponding word vectors. The German STTS POS-Tags were computed with the Marmot POS-Tagger (Müller et al., 2013). We furthermore compute 30 dimensional character embeddings on the shared task data (i.e., not pre-trained), using an LSTM with 50 hidden
units. Dropout is set to 0.2 for the BLSTM and the batch size is set to 50 for all experiments. All models were trained with 100 hidden units.

**Results:** Since the evaluation tool provided by the task organizers always requires a category for computing the scores on Task D, we evaluated our systems using the macro F1 score on the BIO tags. For comparison with the baseline, we compute the score by converting the predictions into BIO format. Table 7 contains our results for Task D.

We trained different set-ups with STL and MTL models. First of all, we evaluated STL against MTL by training two models on 50 dimensional Word2Vec embeddings. For the MTL set-up, we defined the BIO tagging (D) as the main task and added tasks A, B, and C as auxiliary tasks. For document-level annotations (Task A and B) each token of the document is tagged with the respective class of the document. As the results show, the MTL set-up did not improve the macro F1 score in this setting. Thus, we tried to improve the predictions of the STL model in our follow-up experiments. The best results were achieved by using 100 dimensional Word2Vec embeddings with additional POS-Tag embeddings. Furthermore, using AdaDelta as an optimizer yielded better results than using Adam.

Further results, using the organizers’ evaluation tool, can be found below.

### 4.3 Task C

**Approach:** There are several difficulties for this task. First, documents may contain several aspects of different categories, making this at least a multi-class classification problem for document-level approaches. Furthermore, in some cases one document contains several aspects of the same category. On a document-level, one either has to give up on predicting multiple aspects of one class, or add classes for each possible combination of categories, leading to a huge number of classes which do not scale well to new data. Second, there exist aspects with NULL targets which are not assigned to any tokens in the text, but still belong to a category and have a polarity. They cannot be expressed properly on a token-level, as they were not annotated with this intention. One solution may be assigning all tokens of a document to a NULL target category, but this leads to overlapping categories on a token-level, adding more difficulty to the task itself.

To obtain aspect category and polarity predictions, we evaluate various combinations of the stacked learner and the sequence tagger. We report the results for three approaches. (1) We use independent STL sequence taggers to predict BIO labeling as well as category and polarity of each token in a document (INDEP). (2) We predict BIO labeling first, and feed each identified entity to our described ensemble model to predict category and polarity of the identified targets (PIPE). (3) We use the Sequence Tagger for BIO tagging and category prediction (label set \( \{B, I, O\} \times \{\text{Allgemein, Sicherheit, ...}\} \)) and the stacked learner for polarity prediction (JOINT).

**INDEP:** We train a separate model for five subtasks, namely the prediction of BIO labeling, category (cat), polarity (pol), NULL category (n-cat), and NULL polarity (n-pol). If the BIO model predicts B or I for a given token, we look up the cat and pol prediction and obtain the final prediction via a majority vote over the span of BI tokens. Since O tokens are mapped to the none class, we only predict category and polarity if both are present. As the n-cat and n-pol predictions do not depend on the BIO prediction, we perform a majority vote over the whole document. Figure 1 shows an example of how we combine the predictions for the individual subtasks from different STL models.

**PIPE:** We train models with the stacked learner for aspect categories and their polarity. As the BIO predictions do not include NULL targets, we train a separate model on the binary task whether or not a document contains a NULL target. Instead, one could also add another class for documents without any aspects, however we decided not to increase the difficulty for Task C as it already contains 20 classes. If a document is predicted to contain a NULL target, it is added as input for category and polarity prediction. Figure 2 shows the interaction of all models and how the predictions are forwarded.

![Figure 1: Combination of predictions from several independent sequence tagging models (Task C).](image-url)
A, B, and D. Table 8 shows the results for our three final systems (INDEP, PIPE, and JOINT) evaluated with the tool provided by the organizers. It calculates the micro F1 scores of only the categories (C-1) and the categories along with their sentiment (C-2) for Task C, and for Task D the micro F1 scores based on exact (D-1) and overlapping (D-2) matching of the offsets. We obtain BIO predictions for Task D using the best model, namely STL_{AdaDelta} (d = 100 + POS-Tags), and trained the additional models for the INDEP and JOINT approaches with the same parameters.

As can be seen, INDEP consistently outperforms the baseline except for C-1. Further, PIPE outperforms INDEP except for D-1, where it performs even worse than the baseline. The JOINT approach lies between INDEP and PIPE on average.

Strangely, the organizers’ evaluation tool includes the category prediction from Task C for calculating the scores of Task D. The reason for this may be a different point of view for Tasks C and D. If one first identifies the targets and predicts the category and sentiment accordingly, the score for Task D should not be affected by the results for Task C. However, if one first predicts all categories and their sentiment in a document and identifies the targets afterwards, it is important to map the targets to their appropriate categories. Then the correct mapping of category and target may be seen as an additional task which has to be considered for calculating the score for Task D. So even if INDEP, PIPE and JOINT have the same BIO output for Task D, their scores differ due to different predictions of category and sentiment. For example, if the sequence tagging model for categories predicts none for a given chunk, the JOINT and INDEP model discard it for the final results, leading to a different score with the provided evaluation tool. While we tried to model the tasks as they were introduced, our approach to first identify the targets and then to predict category and sentiment seems more intuitive. This way, we do not have the problem of dealing with multiple assignments of one category for a document, as the task is solved on a token-level with a distinct label.

Discussion: All three approaches fail to predict any of the categories Design, Image, and QR-Code. In addition, the JOINT model did not predict any of the categories Gastronomisches Angebot, Toiletten, Reisen mit Kindern, and Gepäck. The INDEP model predicted the least number of different cate-

to the next model for prediction.

JOINT: Here, we use the BIO and category predictions of the sequence tagger, while using the stacked learner for polarity prediction. For NULL targets, we train a separate model for category prediction on the stacked learner similar to the PIPE approach. The model is illustrated in Figure 3.

Results: We train the stacked learner using 500 dimensional Word2Vec embeddings, which showed a good performance for tasks A and B. We do not use Sent2Vec or SIF, since many targets for category and polarity prediction consist of only one word. For targets of longer sequences, we average the word vectors over all tokens.

We used the same hyper-parameters as in Tasks
Table 8: Task C and D results calculated with the provided evaluation tool

<table>
<thead>
<tr>
<th></th>
<th>C-1</th>
<th>C-2</th>
<th>D-1</th>
<th>D-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>0.477</td>
<td>0.334</td>
<td>0.244</td>
<td>0.329</td>
</tr>
<tr>
<td>INDEP</td>
<td>0.429</td>
<td>0.377</td>
<td><strong>0.253</strong></td>
<td>0.364</td>
</tr>
<tr>
<td>PIPE</td>
<td>0.476</td>
<td><strong>0.381</strong></td>
<td>0.233</td>
<td>0.386</td>
</tr>
<tr>
<td>JOINT</td>
<td>0.443</td>
<td>0.367</td>
<td>0.250</td>
<td>0.377</td>
</tr>
</tbody>
</table>

5 Conclusion

We presented our submissions to the GermEval 2017 Shared Task, which focused on the analysis of customer feedback about the Deutsche Bahn AG. We used neural sentence embeddings and an ensemble of classifiers for two sub-tasks as well as state-of-the-art sequence taggers for two other sub-tasks. We substantially outperformed the baseline particularly for Task B, the detection of sentiment in customer feedback, as well as for Task D, the extraction of phrases which carry category and polarity of a meaningful aspect in customer feedback.

Acknowledgments

This work has been supported by the German Federal Ministry of Education and Research (BMBF) under the promotional reference 03VP02540 (ArgumenText) and 01UG1416B (CEDIFOR).

References


Fasttext and Gradient Boosted Trees at GermEval-2017 on Relevance Classification and Document-level Polarity

Leonard Hövelmann
University of Applied Sciences and Arts Dortmund (FHDO)
Department of Computer Science
Emil-Figge-Straße 42
44227 Dortmund, Germany
adesso AG
Stockholmer Allee 20
44269 Dortmund, Germany
leonard.hoevelmann@adesso.de

Christoph M. Friedrich
University of Applied Sciences and Arts Dortmund (FHDO)
Department of Computer Science
Emil-Figge-Straße 42
44227 Dortmund, Germany
christoph.friedrich@fh-dortmund.de

Abstract

This paper describes the submissions to the Shared Task on Aspect-based Sentiment in Social Media Customer Feedback for the GermEval 2017-workshop for the two subtasks Relevance Classification (task A) and Document-level Polarity (task B). For each subtask, the results of the same three systems were submitted: a fastText classifier, enhanced with pretrained vectors, gradient boosted trees (GBTs) trained on bag-of-words (BOWs), and an ensemble of GBTs, respectively trained on word embeddings and on BOWs. For the subtask Relevance Classification, the best system yields a micro-averaged $F_1$-score of 0.895 on the dev set. For the subtask Document-level Polarity, the best system achieves 0.782 on the test set. The proposed system achieved the second place out of twelve systems submitted by seven teams for task A for both test sets. For task B, the proposed system achieved the first place for test set one and the second place for test set two out of 17 systems submitted by eight different teams.

1 Introduction

Customer feedback in social networks is a valuable resource for improving the service of companies. Customers often propose improvements and show points of criticism companies were unaware of. It is important to get an impression of the opinions customers hold with regards to companies. Separating relevant and irrelevant feedback requires expensive manual work. The GermEval 2017 Shared Task on Aspect-based Sentiment in Social Media Customer Feedback workshop (Wojatzki et al., 2017) addresses the automatic processing of German language customer feedback regarding its different characteristics. Therefore, four subtasks were defined: binary classification of whether a feedback is relevant to a given instance (e.g. a company) (task A), categorization of the feedback into sentiment classes (positive, neutral, and negative) (task B), binary classification of a specific aspect into sentiment classes (positive and negative) (task C), and opinion target extraction (task D). In order to elaborate the different subtasks, a training set of customer feedbacks on the German railroad company Deutsche Bahn was provided. For each subtask, corresponding class labels to the feedback texts were provided. The goal was to develop a classifier with a high micro-averaged $F_1$-score on the class prediction.

Word embeddings trained with the objective to predict sentiment polarity were successfully applied at SemEval 2014 workshop (Tang et al., 2014a; Tang et al., 2014b). Ensembles of convolutional neural networks (CNNs) and long short-term memories (LSTMs), trained on pretrained word embeddings achieved state-of-the-art performance at SemEval 2017 workshop (Cliche, 2017). Zhang et al. presented character-level CNNs, trained on one-hot encoded character vectors for text classification tasks (Zhang et al., 2015). Gradient boosted trees (GBTs) (Friedman, 2001) have shown good results in a variety of classification tasks. 17 out of 29 systems that have been published on Kaggle’s blog during 2015 used XGBoost, a framework that implements GBTs (Chen and Guestrin, 2016).
The following sections describe the work on two out of the four given subtasks (task A and task B).

2 Dataset

The dataset for the classification task consists of customer feedback texts collected from various social media platforms, the hyperlink to the reviews, and the annotations of class labels per subtask. The class labels contain information about the document-level sentiment polarity (positive, neutral, or negative), the binary relevance label (denoting whether the text contains feedback about the Deutsche Bahn), as well as annotations for other subtasks which will not be considered in this paper.

The data set was split into a training set (train) and a validation set (dev). Two test sets (test) without class labels were provided at a later stage.

<table>
<thead>
<tr>
<th>Dataset</th>
<th># Reviews</th>
<th>Relevance</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>#</td>
<td># true</td>
</tr>
<tr>
<td>train</td>
<td>19449</td>
<td>16217</td>
</tr>
<tr>
<td>dev</td>
<td>2375</td>
<td>3232</td>
</tr>
<tr>
<td>test</td>
<td>4408</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 1: Number of Reviews in the Data Sets and their Respective Relevance Classes

<table>
<thead>
<tr>
<th>Dataset</th>
<th># Reviews</th>
<th>Document-level Polarity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>#</td>
<td># positive</td>
</tr>
<tr>
<td>train</td>
<td>19449</td>
<td>1179</td>
</tr>
<tr>
<td>dev</td>
<td>2375</td>
<td>149</td>
</tr>
<tr>
<td>test</td>
<td>4408</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 2: Number of Reviews in the Different Document-level Polarity Classes

The distribution of the feedbacks over the different classes are shown in Tables 1 and 2. The classes are not equally distributed, neither in the relevance classification subtask nor in the document-level sentiment polarity classification subtask.

3 Text Preprocessing

The review texts were adopted as feature representation and the hyperlinks were not considered. Each review text was tokenized to extract single terms using whitespaces, percents signs, forward slashes, and plus signs as delimiters.

The largest source of training data are Tweets (≈ 48%), followed by Facebook posts (≈ 22%). Therefore, special attention was paid to Twitter-specific text preprocessing. Frequently occurring Twitter usernames related to the Deutsche Bahn like @DB_Bahn, @Bahnansagen, or @DB_Info are pooled by replacing them with ⟨⟨⟨tokendbusername⟩⟩⟩. Other terms containing an “@” are replaced with the token ⟨⟨⟨tokentwitterusername⟩⟩⟩. The terms S-Bahn and S Bahn are replaced with sbahn.

As the fastText classifier removes punctuation marks, the emoticons “:-)”, “:)”, and “;-)”) are replaced by the token ⟨⟨⟨tokenhappysmiley⟩⟩⟩. The emoticons “:-D” and “xD” are replaced by ⟨⟨⟨tokenlaughingsmiley⟩⟩⟩. The emoticons “:-(” and “:;” are replaced by ⟨⟨⟨tokensadsmiley⟩⟩⟩. Punctuation characters are removed. An exception is two or more repetitions of question marks, exclamation points, and periods. These are replaced with the tokens ⟨⟨⟨tokenstrongquestion⟩⟩⟩, ⟨⟨⟨tokenstrongexclamation⟩⟩⟩, and ⟨⟨⟨tokenannoyeddots⟩⟩⟩ in order to retain the emotion, expressed by the usage of such a writing style.

Many of the feedbacks contain time specifications, for example in the following tweet:

Nach 25 Minuten ist mein Nebenschwitzer in der Bahn ausgestiegen.

These time specifications are replaced by ⟨⟨⟨tokenimeexpression⟩⟩⟩ using a regular expression. Money amounts are replaced by ⟨⟨⟨tokenmoneyamount⟩⟩⟩. Other numbers are replaced by ⟨⟨⟨tokennumber⟩⟩⟩. Furthermore, hyperlinks occurring within the text are also replaced by the token ⟨⟨⟨tokenhyperlink⟩⟩⟩ in order to prevent overfitting. Finally, quotation marks are replaced by ⟨⟨⟨tokenquotation⟩⟩⟩.

After grouping related terms, the remaining text is transformed to lower case and stemmed using the German stemmer in Snowball. The character ß is replaced by ss and ä, ö, and ü are replaced by a, o, and u.

In the next step, the feedbacks are vectorized using the hashing trick (Weinberger et al., 2009). For computational efficiency, the resulting vectors are reduced to a length of 16,384 features applying a modulo operation. The vector entries are the TF-IDF (term frequency - inverse document frequency) values of the terms (Spärck Jones, 1972).

1available from: http://snowballstem.org/, last access: 2017-07-19, license: 3-clause BSD
4 Additional Features

The TF-IDF vectors are enriched with additional information from three feature sources: LIWC-features, word defectiveness, and sentiment lexicon. The LIWC-features (Linguistic Inquiry and Word Count) are determined using the German version of the LIWC computer program (Tausczik and Pennebaker, 2010; Wolf et al., 2008), that “counts words in psychological meaningful categories” (Tausczik and Pennebaker, 2010). The LIWC tool creates 93 decimal values, that can directly be used as features.

In addition to the LIWC-features, the vector is augmented with a feature set expressing the belonging of a review to a certain cluster of reviews. To generate these features, the vector space of the fastText word representations was clustered into 100 clusters. In order to determine the cluster centers, a large vector space model was trained on a snapshot of all German Wikipedia articles, a monolingual news corpus, and the feedback texts themselves (train+dev+test). The k-means algorithm was trained on this Euclidean vector space with the objective to determine 100 cluster centers. Binary features resulting from SentiWS (Goldhahn et al., 2012) were used. Therefor, an n-dimensional vector was created, where n is the sum of positive and negative words. Each index in the vector is connected to one word in SentiWS. During transformation, the respective cell in this vector is set to 1 if the word is contained in the review and to 0 otherwise. The list of words considered as positive has a length of 17,626 words and the list of negative words has a length of 19,961 words, resulting in 37,587 additional features.

Finally, a feature expressing word defectiveness was created. For this purpose, the German version of LanguageTool was used. LanguageTool is able to detect a variety of faulty language, including grammatical errors, missing punctuation or wrong capitalization. Since the text provided is lower cased and free from punctuations, only errors that match the GermanSpellerRule were considered, which detects spelling errors in German language. The feature was created by dividing the number of times the GermanSpellerRule matches in a feedback by the number of words in the respective feedback.

5 Feature Selection

The top-1000 features are chosen from the 16,384 hashed BOW features and the additional features except for the SentiWS-features. The features were selected performing a $\chi^2$-test (Liu and Setiono, 1995, pp. 36,37). Tables 3 and 4 show the top 20 features for the sentiment class and the relevance class, applying $\chi^2$-feature selection and mutual-information feature selection to the BOW features. The additional features were not considered for the creation of these tables. The features in Tables 3 and 4 give an impression on how differently $\chi^2$-selection and mutual information selection differently select top features. Since using the hashing trick causes hash collisions, the top features in the tables were determined without using the hashing trick.

The mutual information analysis results in many stop words like articles or pronouns, whereas the $\chi^2$-analysis yields more meaningful terms like streik (strike) or verspat, which is the stem of Verspätung (delay).

<table>
<thead>
<tr>
<th>$\chi^2$ (relevance)</th>
<th>$\chi^2$ (document-level polarity)</th>
</tr>
</thead>
<tbody>
<tr>
<td>gestartet</td>
<td>franzos</td>
</tr>
<tr>
<td>kehrt</td>
<td>streikend</td>
</tr>
<tr>
<td>asteroid</td>
<td>notrufsyst</td>
</tr>
<tr>
<td>germanwing</td>
<td>schnell</td>
</tr>
<tr>
<td>schweiz</td>
<td>aufatm</td>
</tr>
<tr>
<td>barbi</td>
<td>grenzuberschreit</td>
</tr>
<tr>
<td>weltmeist</td>
<td>bahnstreik</td>
</tr>
<tr>
<td>stellenangebot</td>
<td>lokfahr</td>
</tr>
<tr>
<td>bahn</td>
<td>storung</td>
</tr>
<tr>
<td>job</td>
<td>sncf</td>
</tr>
<tr>
<td>cavendish</td>
<td>regionaldirektion</td>
</tr>
<tr>
<td>lik</td>
<td>tarifkonflikt</td>
</tr>
<tr>
<td>oesterreich</td>
<td>schlichtung</td>
</tr>
<tr>
<td>⟨⟨⟨db_username⟩⟩⟩</td>
<td>funkloch</td>
</tr>
<tr>
<td>ad</td>
<td>beend</td>
</tr>
<tr>
<td>anna</td>
<td>paris</td>
</tr>
<tr>
<td>schwebahn</td>
<td>verspat</td>
</tr>
<tr>
<td>ors</td>
<td>gdl</td>
</tr>
<tr>
<td>bigg</td>
<td>streik</td>
</tr>
<tr>
<td>lufthansa</td>
<td>beendet</td>
</tr>
</tbody>
</table>

Table 3: Top Features Using $\chi^2$-Selection

6 Classifiers

The first system (FHDO GBT BOW) are GBTs (Friedman, 2002) on BOW vectors. For this system, hashed TF-IDF weights of individual terms are merged with LIWC-features. Using the fea-
Table 4: Top Features Using Mutual Information-Selection

<table>
<thead>
<tr>
<th>MI (relevance)</th>
<th>MI (document-level polarity)</th>
</tr>
</thead>
<tbody>
<tr>
<td>es</td>
<td>fur</td>
</tr>
<tr>
<td>nicht</td>
<td>den</td>
</tr>
<tr>
<td>zu</td>
<td>im</td>
</tr>
<tr>
<td>re</td>
<td>es</td>
</tr>
<tr>
<td>im</td>
<td>auf</td>
</tr>
<tr>
<td>von</td>
<td>ich</td>
</tr>
<tr>
<td>den</td>
<td>re</td>
</tr>
<tr>
<td>fur</td>
<td>nicht</td>
</tr>
<tr>
<td>ist</td>
<td>das</td>
</tr>
<tr>
<td>auf</td>
<td>mit</td>
</tr>
<tr>
<td>das</td>
<td>ist</td>
</tr>
<tr>
<td>mit</td>
<td>⟨⟨⟨db_username⟩⟩⟩</td>
</tr>
<tr>
<td>⟨⟨⟨hyperlink⟩⟩⟩</td>
<td>ein</td>
</tr>
<tr>
<td>in</td>
<td>⟨⟨⟨hyperlink⟩⟩⟩</td>
</tr>
<tr>
<td>und</td>
<td>⟨⟨⟨number⟩⟩⟩</td>
</tr>
<tr>
<td>⟨⟨⟨number⟩⟩⟩</td>
<td>und</td>
</tr>
<tr>
<td>der</td>
<td>die</td>
</tr>
<tr>
<td>die</td>
<td>der</td>
</tr>
<tr>
<td>bahn</td>
<td>bahn</td>
</tr>
</tbody>
</table>

Table 4: Top Features Using Mutual Information-Selection

ture selection algorithm, top-1000 features were extracted. For the document-level sentiment polarity classification, the top-1000 features include six LIWC features: body, netspeak, other punctuation, positive emotion, auxiliary verbs, and comparisons.

For relevance classifications, no LIWC features are among the top-1000 features. GBTs are trained on these top-1000 features. The maximal depth of the trees was set to 10, and the number of iterations to 30. For document-level sentiment polarity classification, a one-vs-rest strategy (Hülsmann and Friedrich, 2007) was used as the implementation of this system only supports GBTs for binary classification. The Gini-coefficient is used for impurity calculation and logistic loss as the loss function. The step size was initialized with 0.1.

The second system (FHDO_FT) is a fastText classifier (Joulin et al., 2017) trained on the preprocessed text. The word stemming was omitted for fastText classification. Generally, the following default configuration was used. The dimensionality of the word vectors was set to 100, the size of the context window was set to five, negative sampling was used for loss computation and the learning rate was initialized with 0.05. Deviating from the default configuration, the word vectors were obtained using the word vectors from the large unsupervised corpus (see section 4).

The supervised fastText algorithm is constructed as follows: instead of predicting a word, the goal is to predict a class (or label) - for example, true or false in the sense of document relevance. Similar to the continuous bag-of-words CBOW model (Mikolov et al., 2013), fastText works with (language-dependent) word embeddings. In addition to the CBOW model, fastText word embeddings also make use of character-level n-grams. In order to represent a document, word embeddings are averaged and word order is discarded. The model is capable of handling sentences with a varying number of words. FastText also uses word-level bigram features in order to retain some information about the word order. The use of bigrams is based on the impact of bigrams on classification accuracy in sentiment analysis (Wang et al., 2012).

The models whose names start with “FHDO” are

\[
- \frac{1}{N} \cdot \sum_{n=1}^{N} y_n \cdot \log(f(B \cdot A \cdot x_n))
\]

where \(x_n\) “is the normalized bag of features of the \(n\)-th document” (Joulin et al., 2017), \(y_n\) the label, \(A\) and \(B\) are weight matrices, and \(f\) is the softmax-function.

The third model is an ensemble of two GBT classifiers, where one classifier is trained on the BOW vectors and the other on the word embedding vectors, merged with the one-hot encoded cluster belonging and the LIWC features (FHDO_GBT_NSMBL).

The baseline results in Table 5 are provided by organizers of the GermEval 2017 workshop. It is a Support Vector Machine (SVM) trained with term frequency and a sentiment lexicon as features.

7 Results

Table 5 shows the results of all three systems. To avoid overfitting on the dev set, 5-fold cross validation was performed for model selection on both the train and the dev set. The final submission was created by training on both sets and applying the trained model on the two test sets. The first column gives the name of the system. The second column shows the average of the 5-fold cross-validation and the respective standard deviation while the third column gives the results of the classifier trained on the given training set and validated on the given dev set.

The models whose names start with “FHDO” are

\[http://bit.ly/2xR0zCi\] [google.com], last access: 2017-07-24
The submitted models described in the previous section. The other models were developed for comparison reasons.

FT_UNPROCESSED is a model applying the same fastText classifier used for FHDO_FT to an unprocessed version of the dataset, where only the tokens were split using whitespaces as delimiters. This model is 1.8 percentage points worse for document-level polarity classification and 1.6 percentage points worse for relevance classification. FT_NO_PRETRAIN is trained as FHDO_FT, preprocessing the text, but without incorporating the pretrained vectors. This model is 1.9 percentage points worse for document-level polarity and .07 percentage points worse for relevance classification. GBT_TOP_1000 are GBTs with the same configuration as the ones in the submissions, that has been trained on the top-1000 TF-IDF features from the feature selection only.

Table 5: Results of submitted systems and baseline. First column: name of the respective system. Second column: average micro-averaged $F_1$-score and the respective standard deviation. Third column: results on the official dev set. Results in the second column were computed using 5-fold cross validation.

<table>
<thead>
<tr>
<th>System</th>
<th>5-fold CV</th>
<th>Train / Dev</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relevance</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Baseline</td>
<td>0.700 (±0.008)</td>
<td>0.710</td>
</tr>
<tr>
<td>FHDO_FT</td>
<td>0.775 (±0.007)</td>
<td>0.782</td>
</tr>
<tr>
<td>FHDO_GBT_BOW</td>
<td>0.728 (±0.006)</td>
<td>0.729</td>
</tr>
<tr>
<td>FHDO_GBT_NSMBL</td>
<td>0.751 (±0.004)</td>
<td>0.754</td>
</tr>
<tr>
<td>FT_UNPROCESSED</td>
<td>0.757 (±0.006)</td>
<td>0.760</td>
</tr>
<tr>
<td>FT_NO_PRETRAIN</td>
<td>0.756 (±0.006)</td>
<td>0.764</td>
</tr>
<tr>
<td>GBT_TOP_1000</td>
<td>0.728 (±0.007)</td>
<td>0.728</td>
</tr>
<tr>
<td>GBT_LT_TOP_1000</td>
<td>0.728 (±0.007)</td>
<td>0.728</td>
</tr>
<tr>
<td>GBT_W2V_ONLY</td>
<td>0.720 (±0.009)</td>
<td>0.727</td>
</tr>
<tr>
<td>GBT_W2V_LIWC</td>
<td>0.718 (±0.008)</td>
<td>0.725</td>
</tr>
<tr>
<td>GBT_W2V_SENLEX</td>
<td>0.734 (±0.002)</td>
<td>0.731</td>
</tr>
<tr>
<td>MLP_TOP_1000</td>
<td>0.734 (±0.005)</td>
<td>0.744</td>
</tr>
<tr>
<td>MLP_W2V_ONLY</td>
<td>0.719 (±0.011)</td>
<td>0.721</td>
</tr>
<tr>
<td>MLP_W2V_LIWC</td>
<td>0.585 (±0.008)</td>
<td>0.587</td>
</tr>
<tr>
<td>MLP_W2V_LIWC_SENLEX</td>
<td>0.646 (±0.007)</td>
<td>0.653</td>
</tr>
<tr>
<td>MLP_W2V_SENLEX</td>
<td>0.731 (±0.006)</td>
<td>0.742</td>
</tr>
</tbody>
</table>

8 Conclusion

The best results for both subtasks have been achieved by applying the fastText classifier with the pretrained vectors on the preprocessed dataset. Preprocessing the text improved the micro averaged $F_1$-score by approximately two percentage points. The result of a fastText classifier applied to the unprocessed text was still better than the other tested classifiers. Using fastText without the pretrained vectors causes a drop of approximately two percentage points for document-level sentiment polarity classification and of 0.7 percentage points for relevance classification. Applying GBT classifiers to BOW representations instead of fastText word embedding representations results in a higher micro-averaged $F_1$-score for both subtasks. Using an ensemble of two different GBT classifiers on two different variants of the dataset yields a model with low variance for both subtasks. From all classifiers, MLPs have shown the poorest performance. Future work should analyze the impact of the different features and perform an error analysis. FastText brought results on a par with other state-of-the-art participating systems. The effect of incorporating subword information for German language with its compound nouns should be further examined.
References


GermEval 2017: Sequence-based Models for Customer Feedback Analysis

<table>
<thead>
<tr>
<th>Pruthwik Mishra</th>
<th>Vandan Mujadia</th>
<th>Soujanya Lanka</th>
</tr>
</thead>
<tbody>
<tr>
<td>IIIT-Hyderabad</td>
<td>i.am+ LLC.</td>
<td>i.am+ LLC.</td>
</tr>
<tr>
<td>Hyderabad, India</td>
<td>Bangalore, India</td>
<td>Singapore - 577177</td>
</tr>
<tr>
<td>500032</td>
<td>560071</td>
<td></td>
</tr>
</tbody>
</table>

Abstract

In this era of information explosion, customer feedback is an important source of published content; especially in the social media. Accordingly, analysis of customer feedback is a necessity for companies and enterprises to understand and adapt based on it. Huge volumes of feedback is tough to peruse manually and hence require automated feedback analysis along with associated polarity detection. Customer feedback analysis forms the theme for GermEval Task, 2017 (Wojatzki et al., 2017). In this paper, we describe our approaches for different subtasks in the GermEval Task 2017. The major part of our method is based on a bi-LSTM neural network with word embeddings as features. We observe that our approaches outperform the baseline system for three of the four tasks. The rationale behind us using a bi-LSTM solution will be presented in subsequent sections.

1 Introduction

Customer feedback analysis plays a very important role for an organization to have a clear overview about its products and offered services. A lot of users are expressing their opinions on social media that amounts to the huge influx of data. Therefore extracting relevant information from the gathered data is highly essential. This shared task intends to achieve the same from customer reviews about Deutsche Bahn, a German public train operator.

The Germeval shared task is divided into four subtasks.

A. Relevance Classification - This subtask deals with the identifying if a feedback is about Deutsche Bahn. Hence, we design this as a binary classification problem.

B. Document-Level Polarity - This subtask is a multi-class classification problem where one needs to identify whether the feedback about Deutsche Bahn is positive, negative or neutral.

C. Aspect-Level Polarity - This subtask involves the identification of all the aspects present in a review. The goal is to identify the category and the sentiment of an aspect. This is a multi-label classification problem where multiple classes can be active at the same time.

D. Opinion Target Extraction - Finally, this subtask identifies all the opinions and extracts them.

Sentences containing feedback require positional intelligence based on linguistics for two of the four tasks: (a) identification of the aspect, (b) polarity associated with the aspect. Positional intelligence can only be derived by learning sequential structure of the language or regular patterns associated with feedback. Bi-LSTMs are known for their sequential learning capacity and hence, we base our solutions on bi-LSTM driven training and prediction for three of the tasks.

We use bidirectional LSTM (Graves and Schmidhuber, 2005) for first three tasks and structured perceptron (Collins, 2002) for the fourth task. The paper is organized as follows: Section 2 lists down the related work and Section 3 describes our approach. Section 4 presents the experiments and results on the development set. Section 5 discusses the performance numbers and Section 6 details
about the error analysis. Section 7 concludes the
paper with possible future work.

2 Related Work

Sentiment polarity identification has always been
an active research area. The polarity label can
range from coarse to fine. The coarse polarities can
be positive and negative while much finer labels
may refer to highly positive, positive, neutral,
negative, highly negative. So far many learning
approaches like Naive Bayes, SVM, Maximum
Entropy classifiers have been employed (Pang
et al., 2002) (Agarwal et al., 2011) to tackle it.
But these approaches rely heavily on custom
word lists, part-of-speech tags, fixed syntactic
pattern (Turney and Littman, 2003) and other
sentential information. (Socher et al., 2013) used
deep recursive models to capture the semantics
for longer phrases which eventually improved
sentiment classification. (Qian et al., 2016) used
linguistically regularized LSTM (Hochreiter and
Schmidhuber, 1997) for sentiment classification
and the effects of intensifiers and negations on
polarity of sentences.

But most of the work has been done on
resource rich English language. For German,
(Waltinger, 2010) translated English dictionaries
to put together a consolidated German lexicon.
This German lexicon achieved good results on
sentiment classification. (Momtazi, 2012) created
a German opinion dictionary with substantial
coverage. They built a rule-based sentiment
identification system for automatic sentiment
detection on social media data.

Aspect or Opinion Target identification deals
with the identifying the target words for which a
sentiment is expressed. The previous approaches
(Jakob and Gurevych, 2010), (Hamdan et al., 2015),
(Kessler and Nicolov, 2009) depend on the part-of-
speech tags, dependency relations between tokens,
distance from the nearest noun phrase for finding
the target words. SVM and CRF have been used
extensively for this task.

3 Approach

Most of the machine learning algorithms require
task specific feature engineering for achieving de-
dpendable results. When it comes to natural lan-
guage processing (NLP), these features depend on
the knowledge of the domain, language or both,
which is a limitation. Hence, for this shared task,
we resisted ourselves from using language specific
features. We describe our approaches for each of
the subtasks in the following subsections. For the
first 3 subtasks, we make use of Keras (Chollet
and others, 2015) deep learning library in-order to
employ multi-layer perceptron(MLP) and bidirec-
tional LSTM (bi-LSTM) (Graves and Schmidhuber,
2005). We used Adam optimizer (Kingma and Ba,
2014) with a learning rate of 0.001.

In sequence encoding/tagging, we have access to
both past and future words for a given time stamp.
The bidirectional LSTM network utilize these con-
textual information in both directions (Graves and
others, 2012). In doing so, we can efficiently make
use of past words (and their features) and future
words (and their features) for a given time stamp.
For the classification tasks (task 1 and 2), we train
bidirectional LSTM networks using back propaga-
tion to learn and remember the forward and back-
ward context and to learn categorical information,
we use softmax layer over the output of the bi-
LSTM network.

3.1 Relevance Classification

In our MLP architecture, we use an additional sen-
timent feature where we look up in a sentiment
lexicon (Waltinger, 2010), each word in a review.
If a given word is present, it is assigned the appro-
priate sentiment, neutral otherwise. Each word is
represented by its distributional vector which we
obtain from a pre-trained Glove word embedding
model (Pennington et al., 2014). In the bidirec-
tional LSTM architecture, the only features we use
to train the system are a sequence of word embed-
dings trained on Glove. We will use Glove vectors
and word vectors interchangeably.

3.2 Document-Level Polarity Identification

The architecture and parameters for this subtask are
similar to that of the earlier one. We did experiment
with part-of-speech tag features and glove vectors.
Spacy POS-Tagger\(^2\) is used for part-of-speech tag-
ging each feedback sentence. For this task, only
glove vectors of adjectives and adverbs are chosen
which are tagged by the Spacy POS tagger.

\(^2\)https://spacy.io/docs/usage/pos-tagging
3.3 Aspect Identification

We modeled this problem as a multi-label classification task where multiple labels can be active at a given time. Mean square error is used as the loss function. Consider an example “Streik endet vorzeitig Der Streik der Gewerkschaft Deutscher Lokomotivführer (GDL) endet am 21.” For this sentence, there are 2 aspects, Allgemein:neutral for the two occurrences of the word “Streik”. We combined an aspect and its corresponding polarity to represent a label. The same kind of architecture is also employed in this task as the above two tasks. In this model, multiple occurrences of the same aspect in a single document can not be captured.

3.4 Opinion-Target Identification

Given that Opinion-Target Identification is a sequence labeling task, we used Conditional Random Fields (Lafferty et al., 2001)3 and structured perceptron (Collins, 2002)4 algorithms for this task. Each word in a sample is represented by its lowercase, combination of prefix and suffix characters, length of the word, length of the sequence, its immediate context. Morphological features for a word are important cues for the identification of its lexical category. As the part-of-speech tags were noisy, we use these features instead. Different combinations of the features have been tested and the best possible combination for this task has been chosen. The target words are annotated in BIO (beginning inside out) format as a preprocessing task before training and testing.5

4 Experiment Setup

The corpus size provided in the shared task is detailed below:-

4.1 Corpus Details

<table>
<thead>
<tr>
<th>Type</th>
<th>No Of Samples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train</td>
<td>19449</td>
</tr>
<tr>
<td>Dev</td>
<td>2375</td>
</tr>
<tr>
<td>Test_TimeStamp1</td>
<td>2566</td>
</tr>
<tr>
<td>Test_TimeStamp2</td>
<td>1842</td>
</tr>
</tbody>
</table>

Table 1: Corpus Size

4.2 Preprocessing

As the data provided for this task consisted of social media text, we tokenized the data as a preprocessing step. As part of the tokenization, we considered punctuations except the α and # as individual tokens. α and # are associated with twitter handles and user ids, so we did not tinker them. The impact of tokenization is detailed below. The train and dev accuracies are reported in terms of percentage:-

<table>
<thead>
<tr>
<th>Model</th>
<th>Train_Acc</th>
<th>Dev_Acc</th>
</tr>
</thead>
<tbody>
<tr>
<td>TaskA with punctuation</td>
<td>99.67</td>
<td>86.90</td>
</tr>
<tr>
<td>TaskA with no punctuation</td>
<td>99.70</td>
<td>87.32</td>
</tr>
<tr>
<td>TaskB with punctuation</td>
<td>98.57</td>
<td>69.81</td>
</tr>
<tr>
<td>TaskB with no punctuation</td>
<td>98.42</td>
<td>69.85</td>
</tr>
</tbody>
</table>

Table 2: Experiments with and without punctuations

As the data provided for this task consisted of social media text, we tokenized the data as a preprocessing step. As part of the tokenization, we considered punctuations except the α and # as individual tokens. α and # are associated with twitter handles and user ids, so we did not tinker them. The impact of tokenization is detailed below. The train and dev accuracies are reported in terms of percentage:-

We can observe that there is very minor improve-
the glove vectors and sentiment vectors as features, the size of the sample is 200∗(100 + 3) = 20600. For MLP, all the vectors are concatenated and presented as an input sample whereas all the vectors are given as a sequence of vectors in case of a bidirectional LSTM. If a document contains more than 200 words, only the first 200 words are used in the experiments and the rest are ignored. When a document has less than 200 words, it has to be padded with zero vectors. We used post-padding for our experiments. Evaluation code has been provided by the users\(^6\). For the structured perceptron, the number of iterations was fixed to 10 with viterbi decoding (Viterbi, 1998).

4.4 Experimental Results on Development Set

The results of the experiments on the development set are detailed in this section. The baseline system was provided by the organizers\(^7\). For TaskB, we did experiments where one network learns with glove vectors and the other learns with sentiment vectors. The individual networks are fully-connected dense networks. The hidden layer merges the outputs of these two networks, and the final layer is a softmax layer to predict the polarity of the document. The results of this experiment is tabulated below.

<table>
<thead>
<tr>
<th>Model</th>
<th>Feature</th>
<th>Train_Acc</th>
<th>Dev_Acc</th>
</tr>
</thead>
<tbody>
<tr>
<td>MLP</td>
<td>glove vectors</td>
<td>96.54</td>
<td>69.22</td>
</tr>
<tr>
<td>MLP</td>
<td>sentiment vectors</td>
<td>82.42</td>
<td>67.83</td>
</tr>
<tr>
<td>MLP</td>
<td>glove and sentiment vectors</td>
<td>97.81</td>
<td>70.11</td>
</tr>
<tr>
<td>MLP</td>
<td>glove vectors of Adjectives and Adverbs</td>
<td>82.3</td>
<td>67.43</td>
</tr>
<tr>
<td>Merging 2 MLPs</td>
<td>one with glove vector and other with sentiment vector</td>
<td>95.59</td>
<td>70.57</td>
</tr>
<tr>
<td>BiLSTM</td>
<td>glove vectors</td>
<td>97.22</td>
<td>72.17</td>
</tr>
</tbody>
</table>

Table 4: TaskB Experiments

<table>
<thead>
<tr>
<th>Model</th>
<th>Feature</th>
<th>F1_Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>MLP</td>
<td>glove vectors</td>
<td>0.44</td>
</tr>
<tr>
<td>MLP</td>
<td>glove &amp; sentiment vectors</td>
<td>0.46</td>
</tr>
<tr>
<td>BiLSTM</td>
<td>glove vectors</td>
<td>0.47</td>
</tr>
</tbody>
</table>

Table 5: TaskC Experiments for Category + sentiment

(category - 0.4208, category+sentiment - 0.3485), (d) Opinion Target Extraction - 1st place (exact_matching - 0.2202, overlap_matching - 0.2208). For the final submitted test results, we used BiLSTM for TaskA, MLP for TaskB, BiLSTM for TaskC and Structured Perceptron for TaskD. All the results are reported in terms of the micro F1-scores.\(^8\)

5 Performance Analysis

Training an MLP model is much faster than a bi-LSTM model. The reason could be the memory component associated with LSTMs. The performance evaluation metrics are shown in Table 8. Due to memory limitation of our system, we could not test for bigger batch sizes.

---

\(^6\)https://github.com/muchafel/GermEval2017
\(^7\)BaseLineSystem-https://github.com/uhh-lt/GermEval2017-Baseline

\(^8\)All the F1-Scores refer to the micro F1-Scores
6 Error Analysis

We could observe from the results that adding sentiment features to the network did not improve the dev-set accuracy. We therefore did not consider this feature in our final model. Furthermore, the POS tag for a word does not help in identification of its polarity. In addition, there was error propagation due to errors caused by the Spacy POS tagger on social media text. Hence, POS tags as features have been ruled out in this scenario.

Bidirectional LSTMs are able to model the input sequences better and therefore resulted in higher classification accuracy than MLPs. MLPs performance is relatively low due to the fact that there is no positional information for an input sequence when compared to LSTMs (Hochreiter and Schmidhuber, 1997). Structured Perceptron and CRFs are performing similarly for TaskD. POS Tags have been ignored as mentioned above. We could observe that a word along with its immediate neighbors plays an important role in identification of the target words. The target words can be of a single word or multiple words. The numbers of words in case of a multi-word target word is either three or four. Therefore, a context window of length 3 performs better than a context window of length 5. The errors in the MLP and bi-LSTM models are due to data sparsity. Better vector representations can be learnt with more data.

In the example sentence, “Von den horrenden Preisen und verwirrenden Zonen mal ganz zu schweigen.”, there are 2 aspects “Preisen” and “Zonen”. The system was unable to predict them as such. The training data has several instances of “Zonen”, but neither of them are target words nor do they have any sentiment words in the context. Hence, the system failed to identify this instance. “Preisen” similarly has several instances in the training data, some of them are aspects and some of them are not. The system incorrectly tags it as non-aspect. Even in the presence of a sentiment word as its neighbor, a word is not an aspect in the training data. These are difficult cases for a system to identify and require better semantic representation of the text.

<table>
<thead>
<tr>
<th>Model</th>
<th>Feature</th>
<th>F1 Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRF</td>
<td>word + length features</td>
<td>0.35</td>
</tr>
<tr>
<td>Structured Perceptron</td>
<td>word + length features</td>
<td>0.38</td>
</tr>
<tr>
<td>CRF</td>
<td>word + length features + cw=3</td>
<td>0.42</td>
</tr>
<tr>
<td>Structured Perceptron</td>
<td>word + length features + cw=3</td>
<td>0.43</td>
</tr>
<tr>
<td>CRF</td>
<td>word + length features + cw=5</td>
<td>0.46</td>
</tr>
<tr>
<td>Structured Perceptron</td>
<td>word + length features + cw=5</td>
<td>0.47</td>
</tr>
<tr>
<td>CRF</td>
<td>word + length features + cw=5</td>
<td>0.41</td>
</tr>
<tr>
<td>Structured Perceptron</td>
<td>word + length features + cw=5</td>
<td>0.42</td>
</tr>
</tbody>
</table>

Table 6: TaskD Experiments

<table>
<thead>
<tr>
<th>SubTask</th>
<th>Baseline</th>
<th>Our System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task1</td>
<td>0.88</td>
<td>0.91</td>
</tr>
<tr>
<td>Task2</td>
<td>0.71</td>
<td>0.71</td>
</tr>
<tr>
<td>Task3-only categories</td>
<td>0.61</td>
<td>0.56</td>
</tr>
<tr>
<td>Task3-categories &amp; sentiment</td>
<td>0.49</td>
<td>0.47</td>
</tr>
<tr>
<td>Task4-exact matching</td>
<td>0.2</td>
<td>0.46</td>
</tr>
<tr>
<td>Task4-overlapping matching</td>
<td>0.28</td>
<td>0.49</td>
</tr>
</tbody>
</table>

Table 7: Comparison of Baseline-System and our system micro F1-Scores

<table>
<thead>
<tr>
<th>Model</th>
<th>Avg Training Time Per Epoch(sec)</th>
<th>#Epochs</th>
<th>BatchSize</th>
</tr>
</thead>
<tbody>
<tr>
<td>MLP</td>
<td>3</td>
<td>50</td>
<td>128</td>
</tr>
<tr>
<td>Bi-LSTM</td>
<td>1800</td>
<td>20</td>
<td>4</td>
</tr>
</tbody>
</table>

Table 8: Performance Evaluation for Models
7 Conclusion & Future Work

In this paper, we describe our systems for all the subtasks. We show that systems can achieve a reasonable accuracy with distributional glove vectors without hand crafted features. We also explore target word identification with structured perceptron and CRF by using just basic prefix and suffix features. We could achieve comparable accuracies with these minimal features.

It is intuitive that specific words in a text influence its overall sentiment. In the future we plan to model this by incorporating attention mechanism coupled with a bi-LSTM model. We plan to use character embeddings for the words whose word embeddings could not be learnt. This might help us improve the overall models and make it robust to minute errors/typos. We also plan to add some linguistic features which are observed in social media text like presence of symbols and numbers in order to improve in target word identification. A German lemma dictionary can be used to replace every word in the vocabulary and its morphological variants with its lemma. A sentiment word is known to be present in the neighborhood of a target word. Hence, a sentiment lexicon can also be used to identify the target words.

Acknowledgements

We thank Silpa Kanneganti for her valuable review and feedback for us on the paper.

References


François Chollet et al. 2015. Keras. https://github.com/fchollet/keras.


Hussam Hamdan, Patrice Bellot, and Frederic Bechet. 2015. Lsislf: Crf and logistic regression for opinion target extraction and sentiment polarity analysis. In SemEval@ NAACL-HLT, pages 753–758.


Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, Christopher Potts, et al. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In Proceedings of the conference on


IDS_IUCL: Investigating Feature Selection and Oversampling for GermEval2017

Zeeshan Ali Sayyed†, Daniel Dakota†‡, Sandra Kübler†
†Indiana University, ‡Institut für Deutsche Sprache
zasayyed@indiana.edu, ddakota@ids-mannheim.de, skuebler@indiana.edu

Abstract

We present the IDS_IUCL contribution to the GermEval 2017 shared task on “Aspect-based Sentiment in Social Media Customer Feedback”. We choose to compete in both subtasks A & B. Our system focuses on handling the imbalance in the data sets, by focusing on feature selection and oversampling of the minority classes. We achieve 0.916 micro-F for relevance (task A) and 0.781 for polarity (Task B) on the development set. For task A, we reach the best scores among the submitted systems, for task B the 5th best results for the synchronic test set and the best result for the diachronic test set.

1 Introduction

Sentiment analysis is a field that has seen much attention in recent years, but most of the research concentrates on English. In comparison to English, there exists little work that has focused on sentiment related problems for German. One of the primary issues has been a lack of resources. However, with the creation of German specific resources, such as the Multi-Layered Reference Corpus for German Sentiment Analysis (Clematide et al., 2012), there has been a growing interest in various areas related to semantics and sentiment analysis with a German specific focus. This has resulted in shared tasks on Name Entity Recognition (Benikova et al., 2014), lexical substitution (Miller et al., 2015) and two tasks extracting subjective expressions, sources, and targets in parliamentary speeches (Ruppenhofer et al., 2014; Ruppenhofer et al., 2016).

In the current paper, we describe the IDS_IUCL system that participated in the GermEval 2017 shared task on “Aspect-based Sentiment in Social Media Customer Feedback” (Wojatzki et al., 2017). We choose to compete in subtasks A & B. Our approach uses both word and character n-grams as features, in combination with feature selection based on Information Gain (IG) and L1-regularization (Lasso). Sampling is then used with Gradient Boost to create a final classifier. We achieve 0.916 micro-F for relevance (task A) and 0.781 for polarity (Task B) on the development set. On the official test sets, we reach micro-F-scores of 0.903 and 0.906 on the synchronic and diachronic test sets respectively for task A and 0.734 and 0.750 on the synchronic and diachronic test sets for task B. For task A, we reach the best scores among the submitted systems, for task B the 5th best result for the synchronic test set and the best result for the diachronic test set.

2 Related Work

Perhaps the most closely related shared task to the current one is the sentiment analysis task on Twitter for English (Nakov et al., 2013). Submitted classifiers were predominantly either SVMs or Naive Bayes using a wide range of features. Work on polarity in German sentiment is still a relatively new area, with most of the current approaches profiting from multilingual resources: Debecke (2008) shows how German documents can be translated into English, with positive or negative sentiment analysis performed using English resources on the translated texts achieving reliable performance. This strategy is also seen in work by Momtazi (2012), who automatically translated an English opinion lexicon into German and assigned fine-grained scores to specific words. The resulting approach is rule-based; it outperforms both an SVM and Naive Bayes approach on detecting positive or negative sentiment in social media posts about German celebrities. In both the above cases, however, the systems utilize existing resources for English and attempt to transfer this knowledge to German. Although this approach has been shown
to be feasible, such strategies require many assumptions to be made about the 1:1 correspondence of meaning between languages, which is often not the case. As a consequence, language specific nuances cannot be captured. For this reason, we have decided to work with a knowledge poor approach, focusing on automatically extracting reliable features directly from the training data.

3 Methodology

3.1 Dataset

There are 21,799 samples in the dataset provided by the shared task. For system development, we divide the dataset into a training and a development set for each subtask, using stratified sampling. The training set consists of 19,431 samples (≈89%) whereas the development set consists of 2,368 samples (≈11%).

The distributions of class labels in the training and development set for the relevance task are shown in Table 1, the ones for the polarity task in Table 2. The distribution of class labels clearly shows that the classes are imbalanced. For this reason, we have decided to work with a knowledge poor approach, focusing on automatically extracting reliable features directly from the training data.

3.2 Data Preprocessing

Since the dataset consists of social media data, the text shows many typical characteristics of this genre. In order to extract maximal information from the comments, we process hashtags and handles, as well as URLs. We remove all #s and @s from the tweets leaving just the attached characters (e.g., #Bahn becomes Bahn). We also replace all identifiable URLs (e.g., http or www) by a single URL token. We do not normalize capitalization due to the role it plays in German orthography. We also do not remove punctuation based on the assumption that punctuation in social media data often carries sentiment. Repeated punctuation, for example, serves as an intensifier. We also assume that if punctuation is not relevant, feature selection will delete those features.

3.3 Feature representation

We extract features from the training set using the standard bag-of-words as well as bag-of-characters approach. We use word unigrams, bigrams, and trigrams as well as character n-grams ranging from strings between three and eight characters in length. This results in a total of 1,099,714 features, clearly showing the necessity for feature selection.

3.4 Feature selection

Before any sophisticated feature selection step, frequency thresholding is necessary on this feature set to remove less frequent features. Such features are potentially noisy and misleading. Thresholding also prevents the classifier from overfitting. We test two different thresholds: The first threshold removes all features occurring ≤50 times in the entire training set, the second threshold is set to 100. The choice of threshold has consequences for the size of the features set: The threshold of 50 reduces the number of features to 18,722 whereas a threshold of 100 reduces it to 11,292.

Kübler et al. (2017) show that feature selection using multiclass IG is effective for multiclass sentiment analysis in English when there exists a high imbalance in the classes. Hence, we use multiclass IG for reducing the number of features. Moreover, for large datasets and exponentially many irrelevant features, logistic regression with L1 regularization (Lasso) has been shown to perform successfully (Ng, 2004). Hence, we perform both feature selection techniques on our dataset and then combine the features sets contributed by the two techniques. We use the implementation available from scikit-learn\(^1\) for logistic regression with L1 regularization. We use Lasso with default parameters.

We choose the top 2,000 features from multiclass IG, along with features with non-zero weights from Lasso. When combining the two feature sets, we eliminate duplicates. The distributions of the

\(^1\)http://scikit-learn.org/stable/
resulting feature sets for the two different tasks are shown in Table 3. Obviously, Lasso adds considerably fewer features than IG. However, these features have a positive effect on system performance (see below). Additionally, we see that there are between 200 and 350 features that are selected by both algorithms. The fact that this number is low shows that the feature selection methods have different biases, and a combination can thus be useful.

### 3.5 Classifier

The official baseline for the subtasks A and B consists of an SVM classifier using term frequency features and an external German sentiment lexicon, which resulted in 38,241 features. We performed initial experiments using an SVM and a random forest classifier with the top 2000 features returned from IG, reaching suboptimal results below the official baseline. In contrast, regularized gradient boosted trees (Chen and Guestrin, 2016) perform well on the problem. We use the implementation available in the XGBoost library. This implementation is known for its scalability and robustness given sparse data. The dataset created by the n-gram models is large but very sparse, and hence shows the characteristics for which extreme gradient boosted machines are well suited. The parameters of the algorithm are tuned to combat the imbalance in the dataset. We performed a non-exhaustive search for parameter settings, optimizing performance on the development set.

### 3.6 Sampling

In order to handle the problem of class imbalance, we perform oversampling of the minority classes, using adaptive synthetic sampling (He et al., 2008) in the implementation provided by the imbalanced-learn toolkit (Lemaitre et al., 2016). The standard method for over-sampling minority examples is the synthetic minority over-sampling technique (SMOTE), which generates artificial samples based on feature space similarities between the existing samples of the minority class. This method can create good synthetic samples if the features are meaningful and densely populated in the search space. However, our feature matrix is extremely sparse. Additionally, SMOTE runs the risk of over-generalization because it generates the same number of synthetic samples for every minority class sample without regard to its neighboring samples. Thus, we chose adaptive synthetic sampling, which mitigates these problems by using a weighted distribution for different minority classes, where more data is generated for minority samples which are harder to classify and less for those that are easy.

### 4 Results

#### 4.1 Official Results

Table 4 shows the official results of our system compared to the next-best and best performing system respectively. Note that we have optimized the settings for task A but because of time constraints, we used the same settings also for task B without further optimization.

Our system reaches the highest results on both test sets for task A. For the synchronic test set, the results are very close to the next system, for the diachronic test set, the difference is more pronounced. For task B, we reach the highest results for the diachronictest set, but only the fifth best results on the synchronic test set. The results show that the combination of feature selection and minority oversampling provides us with a robust system that can handle shifts in topics over time.

#### 4.2 Results on the Development Sets

We show the results of the IDS_IUC system on the development set in Table 5. We report the official baseline of the shared task as well as our own baseline, established using 2,000 features obtained when using IG and the SVM classifier. Our baseline corresponds to majority classification baseline since the SVM classifier chooses the majority...
Table 5: Results (micro-averaged F1) on the development set

<table>
<thead>
<tr>
<th>Task/Class</th>
<th>off. baseline</th>
<th>SVM baseline</th>
<th>XGboost-50</th>
<th>XGboost-100</th>
<th>XGboost-50+sampling</th>
</tr>
</thead>
<tbody>
<tr>
<td>relevance (A)</td>
<td>0.882</td>
<td>0.815</td>
<td><strong>0.916</strong></td>
<td>0.915</td>
<td>0.916</td>
</tr>
<tr>
<td>polarity (B)</td>
<td>0.709</td>
<td>0.689</td>
<td><strong>0.781</strong></td>
<td>0.777</td>
<td>0.763</td>
</tr>
</tbody>
</table>

Table 6: Number of features per subtask (threshold=50)

<table>
<thead>
<tr>
<th>Task</th>
<th>No. of Feats.</th>
</tr>
</thead>
<tbody>
<tr>
<td>relevance</td>
<td>2 208</td>
</tr>
<tr>
<td>polarity</td>
<td>2 210</td>
</tr>
<tr>
<td>intersection</td>
<td>191</td>
</tr>
</tbody>
</table>

Table 7: Selected overlap and distinct features for each feature selection method (threshold=50)

<table>
<thead>
<tr>
<th>Feat. Selection</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>IG+Lasso Overlap</td>
<td>Bahn, ahn, Zug, Ticket, deshalb, öfter, morgens, hass, mfra, Stre, reik, verk</td>
</tr>
<tr>
<td>IG Relevance</td>
<td>offenbar, Rückfahrt, Fenster, zumindest, Erg, Ank, tark, ohnt</td>
</tr>
<tr>
<td>IG Polarity</td>
<td>Jahren, Hauptbahnhof, Tarifkonflikt, Klimaanlage, lich, chen, unde, ahme</td>
</tr>
<tr>
<td>Lasso Relevance</td>
<td>Bar, Spaß, Fahrer, Bank, läer, örse, letz</td>
</tr>
<tr>
<td>Lasso Polarity</td>
<td>steigen, Kunden, Danke, Stunden, eei, bequ, önt, tehe</td>
</tr>
</tbody>
</table>

First we examine the overlapping features between the relevance and the polarity tasks. In total, there are 191 overlapping features between the two tasks. The first row of table 7 shows examples of such overlapping features. Many of these features are to be expected, e.g., Bahn (Eng. train) and substrings of words such as Stre, most likely deriving from Streik (Eng. strike) and/or Strecke (Eng. route). The minimal overlap is interesting in and of itself given that the base feature set is the same for both tasks. Additionally, there is a sizable number of examples of long n-grams with overlapping substrings between the tasks. I.e., the tasks select similar information, but with a slightly different emphasis. For example, die Strecke (Eng. the route [nom/acc]) is selected for relevance but der Strecke (Eng. the route [gen/dat]) is selected for polarity. Such differences may be caused by data sparsity or by minimally different feature selection values. However, in our cases, those features are not close to the cut-off IG. Thus, they must be indicative of certain structures or contexts that can be associated more strongly with one specific task but not the other. This can be seen when examining some of the selected n-grams, which seem to benefit a particular subtask. For example, schneller (Eng. faster) is only selected for the polarity task, but not for the relevance task while Rückfahrt (Eng. return trip) is selected for relevance. This can be interpreted assuming that being fast is good for a train, thus evoking sentiment, but it may not help with relevance. A return trip, in contrast is a more general term that may be more helpful in determining relevance, but it does not evoke sentiment.

We also examine features chosen when using IG or Lasso in table 7. For the polarity n-grams, there are recognizable features, such as Tarifkonflikt (Eng. trade dispute). For all methods, one can identify possible words for the returned character level n-grams, although they may not be initially intuitive. However, the main impression is that IG selects common features while Lasso seems to prefer less common features that are highly indicative of a class.

5 Further Analysis

5.1 Feature Analysis

We manually examine the selected features to see what types of unique features are returned by IG and Lasso for both relevance and polarity (threshold=50, no sampling).
### Table 8: Additional results for relevance task

<table>
<thead>
<tr>
<th>Classifier</th>
<th>micro-F</th>
<th>macro-F</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>XGboost-50</td>
<td>0.916</td>
<td>0.847</td>
<td>0.852</td>
<td>0.927</td>
</tr>
<tr>
<td>XGboost-100</td>
<td>0.914</td>
<td>0.838</td>
<td>0.890</td>
<td>0.918</td>
</tr>
<tr>
<td>XGboost-50+sampling</td>
<td>0.916</td>
<td>0.847</td>
<td>0.850</td>
<td>0.927</td>
</tr>
</tbody>
</table>

### Table 9: Additional results for the polarity task

<table>
<thead>
<tr>
<th>Classifier</th>
<th>micro-F</th>
<th>macro-F</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>XGboost-50</td>
<td>0.781</td>
<td>0.625</td>
<td>0.774 0.784 0.732</td>
<td>0.413 0.953 0.351</td>
</tr>
<tr>
<td>XGboost-100</td>
<td>0.777</td>
<td>0.615</td>
<td>0.767 0.779 0.758</td>
<td>0.392 0.956 0.338</td>
</tr>
<tr>
<td>XGboost-50+sampling</td>
<td>0.777</td>
<td>0.620</td>
<td>0.760 0.780 0.785</td>
<td>0.397 0.954 0.345</td>
</tr>
</tbody>
</table>

### 5.2 Alternative Evaluation Metrics

We also have a closer look at the performance of our classifiers using macro-F along with micro-F. Macro-F first calculates the F-score per class and then averages over the classes, which gives equal weight to minority classes rather than giving equal weight to each example, thus making minority classes less important in the evaluation. We also report precision and recall for individual classes. Table 8 shows the results for the relevance task, table 9 for polarity. These results show that the higher threshold results in higher precision for the minority classes (false for relevance and positive for polarity). The oversampling method reaches the highest precision for the minority class for polarity. Further investigation is required to determine why this effect is not evident in the relevance task.

### 6 Conclusion

The IDS JUCL system reaches good results when using XGboost while first experiments with SVM or random forest classifiers resulted in majority classification. We show that feature selection is extremely important for tasks in sentiment analysis with an imbalanced class distribution. We reach our best results with a feature threshold of 50; a higher threshold results in higher precision for the minority class. Oversampling of minority class examples increases precision in the polarity task.

### References


Abstract

This paper presents a hybrid approach to document-level polarity classification of social media (SM) posts. In contrast to previous related works, which relied on sentiment classifiers with either manually designed or automatically induced features, our system simultaneously harnesses both of these options, comprising an SVM module with user-specified attributes and a bidirectional LSTM network whose input embeddings are learned completely automatically. While doing prediction, we unite the decisions of both of these classifiers into a single score vector by taking the sum of their probability estimates and choosing the class with the highest joint value. We evaluate our ensemble on subtask B of GermEval-2017 shared task competition (Wojatzki et al., 2017), getting third place among all competing systems and reaching 0.745 and 0.718 micro-averaged $F_1$ on timestamps one and two of this dataset respectively.\footnote{The source code of our implementation is available online at \url{https://github.com/WladimirSidorenko/CGSA}}

1 Introduction

With the ever increasing role of social media in people’s everyday life, web forums, Facebook walls, and Twitter threads gradually become our primary channels for staying in touch with friends, sharing important information, or just expressing our personal opinions and beliefs. The last purpose is of particular interest to many private companies and organizations, as it turns social networks into an invaluable source of critical market information, providing deeper insights into the general sales atmosphere and revealing wishes, complaints, and preferences of particular customer groups.

Unfortunately, manually analyzing the avalanche of users’ posts can hardly be done rapidly and is impossible to do in a flash. Consequently, any serious economic endeavor nowadays depends on the existence of a reliable automatic sentiment analysis tool. To meet this need, a plethora of different rule-based, machine- and deep-learning methods have been proposed in literature in the last few decades (Pang and Lee, 2008; Liu, 2012), with each of them claiming superior results in comparison to its predecessors.

In this paper, we are going to investigate whether the recent “tornado”-like popularity of deep neural networks for coarse-grained sentiment analysis (Manning, 2015) is indeed backed by empirical results and whether these systems can actually outperform traditional machine-learning methods, which are commonly considered to be the previous state of the art for this task. Furthermore, since these techniques not necessarily need to be in contradiction, we also check whether uniting both approaches into a single ensemble can further improve the quality of the analysis.

We perform our experiments on Subtask B (document-level polarity prediction) of GermEval-2017 (Wojatzki et al., 2017), presenting its dataset in Section 2 of this paper. After shortly describing initial preparation steps in Section 3, we present both methods (SVM and bidirectional LSTM), sketching their features, training mode, and architecture. In the final step, we estimate the effects of different feature groups and hyperparameters on the net results of these systems, concluding and summarizing our findings in the last part of this submission.

2 Data

To train and evaluate our approaches, we downloaded the training and development sets of the GermEval-2017 shared task competition (Wojatzki et al., 2017), getting a total of 23,525 messages. A
detailed breakdown of the the number of posts and class distributions in each of these sets is provided in Table 1.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Total</th>
<th>Positive</th>
<th>Negative</th>
<th>Neutral</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training Set</td>
<td>20,941</td>
<td>14,497</td>
<td>5,228</td>
<td>1,216</td>
</tr>
<tr>
<td>Development Set</td>
<td>2,584</td>
<td>1,812</td>
<td>617</td>
<td>155</td>
</tr>
</tbody>
</table>

Table 1: Class distribution in the GermEval dataset.

As we can see from the statistics, the neutral class distinctly dominates the complete corpus, making up 70% of the training instances. Negative posts form the second most frequent group, accounting for \( \approx 24\% \) of the data; while the positive class represents an absolute minority of the dataset, constituting only 6% of all cases.

Besides a conspicuous imbalance of the polarity classes, an additional challenge of this shared task is posed by the heterogeneity of the provided content: Instead of concentrating on just one popular social channel—as it was done, for example, in similar shared tasks (Nakov et al., 2013; Rosenthal et al., 2014; Rosenthal et al., 2015)—the organizers covered a wide range of possible domains, providing data from 2,291 different sources.

### 3 Preprocessing

We preprocessed all retrieved messages using the rule-based normalization procedure of Sidarenka et al. (2013). In particular, during this step, we split each user post into sentences and tokens, restored some common colloquial spelling mistakes (e.g., “kannste” → “kannst du”, “laaangen” → “lagen”, “vlt” → “vielleicht”), and replaced frequent SM-specific entities (@-mentions, links, e-mail addresses, and emoticons) with unique artificial tokens representing their lexical class (“%User”, “%Link”, “%Mail”, “%PosSmiley”, “%NegSmiley”, etc.). A sample output of this normalization is shown in Example 5.2.

#### Example 3.1

**Original:**

Wochenende! Und der Telekom-Hotspot performt wieder wie tote Füße ... (@DBLounge - @db_bahn) https://t.co/dlIMnKsnLh

**Normalized:**

Wochenende! Und der Telekom-Hotspot performt wieder wie tote Füsse ... (%User - %User) %Link

In the final step of this procedure, we obtained lemmas and part-of-speech tags of the analyzed tokens using the TREE TAGGER of Schmid (1995).

### 4 Method

Once the data were preprocessed, we trained two independent classification systems and united their output at the end into a single ensemble.

#### 4.1 SVM

The first of these systems—a multi-class SVM classifier—was largely inspired by the work of Mohammad et al. (2013). Similarly to these authors, we used a wide variety of different features, which, for simplicity, can be grouped together as follows:

- **punctuation features**, which included the number of repeated exclamation and question marks as well as the count of contiguous sequences of exclamation and question characters;

- **character-level features**, which comprised character n-grams of length three to five as well as the number of capitalized words and words with elongated vowels (e.g., “sooooo”, “guuuut”, etc.);

- **word-level features**, which included contiguous and non-contiguous (i.e., with one of the tokens replaced by a wildcard) sequences of one to four tokens as well as the counts of user mentions, emoticons, and hashtags found in the message;

- **part-of-speech features**, which reflected the number of occurrences of each particular part-of-speech tag in the analyzed instance;

- **lexicon features**, which, similarly to Mohammad et al. (2013), were subdivided into manual and automatic ones;

  - **manual lexicon features** were estimated using the SentiWS lexicon of Remus et al. (2010) and Zurich Polarity List of Clematide and Klenner (2010). For each of these resources and for each of the non-neutral polarity classes, we computed 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 lexicon features were computed using several automatically induced polarity lists. In particular, we re-implemented the dictionary-based approaches of Blair-Goldensohn et al. (2008), Hu and Liu (2004), and Kim and Hovy (2004), applying these systems to GERMANET 9.0 (Hamp and Feldweg, 1997). Another sentiment lexicon was produced using the Ising-spin method of Takamura et al. (2005), for which we again used GERMANET 9.0, extending its data with corpus collocations gathered from the German Twitter snapshot (Scheffler, 2014). Moreover, we also came up with two own additional solutions, in which we derived new polarity lists by performing k-nn and projection-based clustering of word2vec embeddings that had been previously pre-trained on the aforementioned snapshot corpus. For each of these resources, for each of the two polarity classes (positive and negative), we produced four features representing the number of tokens with non-zero scores, the sum and the maximum of all respective lexicon values for all message tokens, and the score of the last term.

To overcome the rigidness of linear decisions in SVM (i.e., the inability of standard SVM to make a boundary between linearly inseparable classes), we split all automatic lexicon features into deciles based on the total range of their observed values, and replaced these attributes with binary features reflecting the respective quantile of their original scores. For example, if a training instance had a feature called hu-liu-positive-sum with the value 15.7, which belonged to the third decile of all seen scores for this attribute, we replaced this feature with the binary attribute hu-liu-positive-sum-3, setting its value to 1. This way, we allowed the separating hyperplane of SVM to be more flexible by having different slopes at different value regions of basically same attributes.

In the last step, we determined the optimal hyper-parameter settings (slack variable $C$) for the classifier using grid search with five-fold cross validation over both training and development data. Afterwards, while preparing the final submission for GermEval, we retrained the system once again on the complete dataset, keeping the slack constant $C$ fixed to its best-performing value.

4.2 Bidirectional LSTM

In addition to the SVM system, we also trained a deep multi-layer bidirectional recurrent neural network. In particular, we used the usual LSTM recurrence (Hochreiter and Schmidhuber, 1997), which took word embeddings $(\vec{w}_1; \ldots; \vec{w}_L) \in \mathbb{R}^{L \times 300}$, where $L$ is the length of the training instance) as input. Following the usual practices for defining such networks, we ran one of the loops from left to right, obtaining an output vector $\vec{y}_L^{+} \in \mathbb{R}^3$ for each sequence position $t$, and let another loop work from right to left, getting a vector $\vec{y}_L^{-} \in \mathbb{R}^3$ for each position. We concatenated the outputs of both recurrences into a single intermediate tensor $\vec{r} \in \mathbb{R}^{L \times 16 \times 2}$, and run a third LSTM loop over its leading dimension ($L$). Eventually, we multiplied the output of this third recurrence $\vec{r}$ at the final step $L$ with a linear transform matrix $W \in \mathbb{R}^{3 \times 16}$, adding a bias term $b \in \mathbb{R}^3$ and applying the softmax non-linearity $\sigma$ to this product:

$$\vec{y} = \sigma \left( W \cdot \vec{r}_L + b \right).$$

We set the initial values of all trained parameters (word embeddings, recurrences, bias terms, and transform matrices) using normal He initialization (He et al., 2015), training the network for five epochs and picking the model which maximized the accuracy on a randomly chosen held-out set. Due to the high imbalance of the target classes (with neutral instances accounting for almost 70% of the training data), we used categorical hinge loss as the primary objective function and optimized it using RmsProp (Tieleman and Hinton, 2012) to speed up the convergence. Last but not least, to prevent overfitting to the training set, we applied Bayesian dropout (Gal and Ghahramani, 2015) to the recurrence loops, setting the Binomial probability of randomly dropping a neuron to 0.2.

4.3 Ensemble

To unite the results of both classifiers, we obtained the decision scores from the SVM system. Since these scores, however, reflected the distance to the separating hyperplanes and could easily outweigh the results of the LSTM module (whose output was guaranteed to be in the range $[0, \ldots, 1]$), we also applied the softmax function to these values. Afterwards, we summed both vectors (the modified
SVM output and the \( \vec{y} \) vector from Equation 1) and chose the class with the maximum total value as the final decision.

5 Evaluation

The results of these systems are shown in Table 2. As we can see from the figures, the SVM approach clearly outperforms the other two options, achieving best results on both timestamps: one, where it attains 0.752 points, and two, where it gets 0.737 \( F_1 \). The second-best system is, not surprisingly, the ensemble, whose micro-averaged results, however, are 0.007 and 0.02 points worse than the respective SVM scores. This degradation is mainly due to the bidirectional LSTM component, which, unexpectedly, yields the worst overall performance (getting 0.727 and 0.704 micro-averaged \( F_1 \)).

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Micro-( F_1 ) Timestamp 1</th>
<th>Micro-( F_1 ) Timestamp 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM</td>
<td>0.752</td>
<td>0.737</td>
</tr>
<tr>
<td>BiLSTM</td>
<td>0.727</td>
<td>0.704</td>
</tr>
<tr>
<td>SVM + BiLSTM</td>
<td>0.745</td>
<td>0.717</td>
</tr>
<tr>
<td>Majority</td>
<td>0.655</td>
<td>0.672</td>
</tr>
<tr>
<td>Random</td>
<td>0.417</td>
<td>0.403</td>
</tr>
</tbody>
</table>

Table 2: Micro-averaged \( F_1 \)-scores on timestamps 1 and 2 of the GermEval test set for Subtask B.

5.1 Qualitative Analysis

Besides calculating these statistics, we also had a closer look at the particular errors made by these classifiers. As it turned out, most of the mistakes on Timestamp 1 were made in the cases when all three classifiers confused negative instances with the neutral class (185 errors). On Timestamp 2, the most frequent type of errors (124 mistakes) were cases when the SVM classifier correctly predicted the neutral class, but the BiLSTM method assigned the negative label with a very high probability, outweighing the former approach in the ensemble decision.

In general, the prevalence of the BiLSTM classifier accounted for the vast majority of ensemble’s errors, with a few examples of this behavior provided below.

**Example 5.1**

**Message:** Zeugen gesucht: 39-Jähriger wurde in S-Bahn von dunkelhäutigen Männern betäubt und überfallen - B.Z. Berlin...

**Gold:** negative

**SVM:** negative

**BiLSTM:** neutral

**SVM + BiLSTM:** neutral

**Example 5.2**

**Message:** Der Bahn-Betriebsrat fordert eine schnelle Einigung im Tarifstreit: %Link gdl-streik bahnstreik

( *The Bahn employee organization demands a quick agreement in the pay dispute: %Link gdl-strike Bahn-strike* )

**Gold:** neutral

**SVM:** positive

**BiLSTM:** negative

**SVM + BiLSTM:** negative

5.2 Comparison with Subtask A

On request of one of the reviewers, we also re-trained our system on Subtask A of GermEval-2017. In this task, we had to predict whether particular posts contained feedback about the “Deutsche Bahn” or not. This way, we hoped to check the generalizability of the proposed methods (i.e., to see whether the same techniques could be equally well applied to other objectives).

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Micro-( F_1 ) Timestamp 1</th>
<th>Micro-( F_1 ) Timestamp 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM</td>
<td>0.816</td>
<td>0.84</td>
</tr>
<tr>
<td>BiLSTM</td>
<td>0.865</td>
<td>0.857</td>
</tr>
<tr>
<td>SVM + BiLSTM</td>
<td>0.873</td>
<td>0.869</td>
</tr>
<tr>
<td>Majority</td>
<td>0.816</td>
<td>0.84</td>
</tr>
<tr>
<td>Random</td>
<td>0.496</td>
<td>0.491</td>
</tr>
</tbody>
</table>

Table 3: Micro-averaged \( F_1 \)-scores on timestamps 1 and 2 of the GermEval test set for Subtask A.

As we can see from the results in Table 3, the answer to this question is negative for the SVM classifier, which performs on a par with the majority class baseline. Nevertheless, the BiLSTM approach is quite promising in this regard, yielding much better figures than for the previous subtask. This time, the ensemble approach outperforms all its single components, mitigating the lower accuracy of SVM.
6 Summary and Conclusions

Summarizing the above findings, we would like to recap that, in this work, we compared three different approaches to coarse-grained sentiment analysis of social media posts: SVM with manually specified features, bidirectional LSTM with automatically induced input representations, and a combination of both. Two of these systems—unfortunately, the weaker ones—were part of the official submission of the PotTS team to the GermEval-2017 competition.

We showed that traditional machine-learning techniques still achieve state-of-the-art results, outperforming modern deep-learning methods. Therefore, we would recommend taking newer DL approaches with a grain of salt, always keeping in mind that the results of machine-learning methods might crucially depend on the specifics of the analyzed data and peculiarities of the task at hand (as we also confirmed in the evaluation section). One potential weakness of these approaches though, is their weak generalizability, which prevents us from reusing them for other (related) objectives.

Acknowledgments

We thank the anonymous reviewers for their helpful suggestions and comments.

References


LT-ABSA: An Extensible Open-Source System for Document-Level and Aspect-Based Sentiment Analysis

Eugen Ruppert‡ and Abhishek Kumar† and Chris Biemann‡

‡Language Technology Group
Computer Science Dept.
Universität Hamburg
http://lt.informatik.uni-hamburg.de

†Indian Institute of Technology Patna
AI-NLP-ML group
Patna, India
http://www.iitp.ac.in

Abstract

This paper presents a system for document-level and aspect-based sentiment analysis, developed during the inception of the GermEval 2017 Shared Task on aspect-based sentiment analysis (ABSA) (Wojatzki et al., 2017). It is a fully-featured open-source solution that offers competitive performance on previous tasks as well as a strong performance on the GermEval 2017 Shared Task. We describe the architecture of the system in detail and show competitive evaluation results on ABSA datasets in four languages. The software is distributed under a lenient license, allowing royalty-free use in academia and industry.

1 Introduction

Sentiment analysis has gained a lot of attention in recent years in the CL/NLP community. Aggregating over the sentiment in a large amount of textual material helps governments and companies to deal with the large increase of user-generated content due to the popularity of social media. Companies can react to upcoming problems and prepare strategies to help users to navigate reviews and to improve their reputation.

While determining document-level sentiment can be framed as a classification task with two or three classes (positive, negative, possibly neutral), identifying and evaluating aspect-based sentiment is more challenging: here, we are not only interested in the polarity of the sentiment, but also to what particular aspect the sentiment refers to – for example people might express in the same product review that they like the high-resolution screen of a phone while complaining about its poor battery life. Aspects are typically classified into a flat taxonomy, and are lexicalized in opinion target expressions (OTEs), which shall be identified by ABSA systems.

Even though a steady number of sentiment analysis tasks have been conducted in the past years on aspect-based as well as other flavors of sentiment analysis, e.g. (Pontiki et al., 2015; Pontiki et al., 2016; Wojatzki et al., 2017), participants mostly do not share their systems, so that others could use or extend them. Even if systems are shared, they are usually not easy to operate, since they typically stay on the level of research software prototypes. A notable exception is Stanford’s CoreNLP project, which however only performs document-level sentiment on English (Socher et al., 2013).

In this paper, we present a fully-featured open-source system for ABSA. Configurations regarding the use of features or the choice of training data can be shared, enabling reproducible results. Our system is flexible enough to support document-level and aspect-based sentiment analysis on multiple languages. Since we also provide feature induction on background corpora as part of the system, it can be applied out of the box.

We focus on engineering aspects. For related work regarding aspect-based sentiment analysis, we refer to the task description papers cited above, as well as recent surveys, e.g. (Medhat et al., 2014).

2 Architecture

The system is designed as an extensible framework that can be adapted to many different datasets. It is able to perform document-level classification as well as the identification of opinion target expressions (OTEs). NLP pre-processing is engineered in the UIMA framework (Ferrucci and Lally, 2004), which contributes to adaptability and modularity. It is a full-fledged system that contains all stages of preprocessing, from reading in different data formats over tokenization to various target outputs, and is aimed at productive use.

1The system is available under the permissive Apache Software License 2.0, http://apache.org/licenses/LICENSE-2.0.html
2.1 Execution and Workflow

The general workflow consists of three major steps (see Figure 1). To prepare model creation, we perform feature induction (1). This step has to be conducted only once when creating a model for a new language or domain. The operator can provide an in-domain corpus to induce features derived from whole-corpus statistics, like tf-idf scores. Furthermore, we support the corpus-informed extension of word lists, such as augmenting a list of positive words with similar words from a background corpus, as described in more detail below. While our system also uses word embeddings as features, their training is not part of our system but needs to be done externally.

In the training step (2), models are trained using labeled training data. The processing pipeline includes readers for several formats to create a document representation, language-specific NLP tools and feature extractors to to create feature vectors. We train machine learning models on these feature representation in order to support two general setups: document-level classification into an arbitrary number of classes, and sequence tagging for extracting spans, such as OTEs.

Finally, the models are used for the classification of new documents (3). This step supports the same file formats and conducts the same feature extraction as in the training step. Additionally, we have included a small web server with an RESTful API with HTML and JSON output (see Listing 1 for an example).

The NLP pipeline includes the rule-based segmenter described in Remus et al. (2016), which allows adapting the tokenization to the target domain, e.g. handle hashtags, cashtags and other types of tokens for social media content. For POS tagging, we rely on OpenNLP\(^2\) for the reason of license compatibility.

3 Features

In this section, we describe our feature induction on background corpora and list the features for

\(^2\)http://opennlp.apache.org/
3.1 Feature Induction

Background Corpus We use an in-domain corpus to induce features and semantic models. E.g., for the background corpus on the GermEval 2017 dataset, we used a web crawl obtained by the language-model-based crawler of (Remus and Bie mann, 2016). If in-domain data is not available, we still recommend to perform feature induction with a background corpus from the same language. On the background corpus, we compute a distributional thesaurus (DT) (Biemann and Riedl, 2013) and a word2vec model (Mikolov et al., 2013) using the according software packages, which are not part of the distribution. However, we provide the models as well as usage instructions on how to compute them. Further, we compute inverse document frequencies (IDF) of words (Spärck Jones, 1973).

Training Data Using the training data and the idf scores, we determine the tokens with the highest tf-idf scores for each document-level class. The top 30 tokens for each class are used as binary features.

Polarity Lexicon Expansion Assuming the existence of a polarity lexicon (e.g. Waltinger (2010) for German), we automatically expand such lexicon for a language using the method described in our previous work (Kumar et al., 2016): First, we collect the top 10 distributionally most similar words for each entry in each polarity class (positive, negative, sometimes also neutral). Then, we filter these expansions by a minimum corpus frequency threshold of 50 in the background corpus. Next, we only keep the expansions that were present in at least 10 of the seed terms. While distributional similarity does not preserve polarity, described aggregation strategy results in a high-precision high-coverage domain-specific polarity lexicon.

For all expansion terms, we calculate the normalized scores for each polarity, resulting in a real-valued weight for each polarity.

3.2 Document-Based Classifier

We use a linear SVM classifier (Fan et al., 2008) for document-based classification. As the feature space is fairly large and sparse (100+K features for GermEval 2017), we can resort to a linear kernel and do not require more CPU-intensive kernel methods.

- **TF-IDF**: We calculate the tf-idf weights for each token using the IDF from the background corpus and the frequency of the token in the current document, using token weights as features. The overall TF-IDF feature vector is normalized with the L2 norm.

- **Word Embeddings**: We use word embeddings of 300 dimensions trained with word2vec (Mikolov et al., 2013) on background corpora. For the document representation, word representation for each word is obtained and then averaged up to get a 300
dimensional feature vector. Word embedding averaging is done unweighted as well as weighted by the token’s tf-idf score. Finally, the averaged feature vector is normalized using the L2 norm.

- **Lexicon:** This feature class allows to supply word lists, recording their presence or absence in a sparse feature vector. We use this feature class for supplying polarity lexicons to our classifier.

- **Aggregated Lexicon:** This feature class also relies on word lists with labels, but aggregates over words from the same class: we supply the relative amount of positive, negative and neutral words in the document, normalized by document length.

- **Expanded Polarity Lexicon:** We use the induced expanded polarity lexicon to generate a low-dimensional feature vector (2-3 features). The expanded polarity lexicon provides a polarity distribution for each term, e.g., schnell (fast) – 0.32 (neg-value) – 0.68 (pos-value). We use this feature by summing up the distributions of the tokens that appear in the expanded lexicon and averaging them.

### 3.3 CRF

The CRF classifier (Okazaki, 2007) is used for annotation of Opinion Target Expressions, cast in a sequence tagging setup. It uses the following symbolic features in the ClearTk\(^3\) framework:

- current token (surface form + lowercased)
- POS tag
- lemma (not available for all languages)
- character prefixes (2–5 characters)
- suffixes (2–5 characters)
- capitalization
- numeric type (identifies types, when numbers are present; e.g. digits, alphanumeric, year)
- character categories (patterns based on Unicode categories)
- hyphenation

These features are computed in a window of +/-2 tokens around the target token.

### 4 Results

In the experimental results reported below, we have used the following background corpora for feature induction: For German, we have compiled a corpus from a focused webcrawal (Remus et al., 2016). For the SemEval tasks, we employ COW (Schäfer, 2016) web corpora\(^4\) for English, Spanish and Dutch.

#### 4.1 GermEval 2017 Shared Task

The GermEval 2017 Shared Task on ABSA (Wojatzki et al., 2017) features a large German dataset consisting of user-generated content from the railway transportation domain. There are four subtasks that cover document-based and aspect-based sentiment analysis. Participants should classify the binary relevance and the document-level sentiment in Subtasks A and B. Next, they should identify aspects in the document and their corresponding sentiment (Subtask C). Finally, OTEs are identified by span and labeled with an aspect and a sentiment polarity in Subtask D. The task features two test sets: documents from the same period as the training data (synchronous) and documents from a later point in time (diachronic). For evaluation, micro-averaged F1 scores are used.

Our system has been developed in the same project that funded the creation of the dataset used in GermEval 2017. Naturally, as the organizer’s entry, it did not compete in the shared task. Nevertheless, we report the ranks our system would have obtained in this task.

Table 1 presents the results on the synchronous dataset and Table 2 on the diachronic dataset. Our system outperforms all baselines and would have ranked highly in the competition, outperforming most submissions on almost every task. On Subtasks A and B, our system is outperformed by a small margin, on Subtasks C and D, we show the best performance overall. We conclude that LT-ABSA is a highly competitive system for sentiment classification on German.

#### 4.2 SemEval-2016 Task 5: Aspect Based Sentiment Analysis

The SemEval-2016 task on aspect-based sentiment analysis (Task 5;Pontiki et al., 2016)) is comparable in structure to Subtasks B, C and D in the GermEval-2017 evaluation. While the overall task was conducted on datasets in eight languages and

[^3]: http://cleartk.github.io/cleartk/

[^4]: http://corporafromtheweb.org/
multiple domains, we have only experimented with the English, Spanish and Dutch datasets. Table 3 presents the results, again with ranks that our system would have obtained in the task. We report scores on Subtask 1, Slots 1 (Sentence-level Aspect Identification) and 3 (Sentiment Polarity), and on Subtask 2, Slot 2 (Document-level Sentiment Polarity). Overall LT-ABSA is able to beat all baselines for the reported slots. Only for SB2, Slot 2 on English, where the baselines rank in the middle, we are outperformed by the baselines. The performance varies across tasks. For the highly contested English datasets, we rank in the lower midfield for SB1 and in the top 5 for SB2. For the less contested Spanish and Dutch datasets, we show a competitive performance.

---

5We used our system out-of-the-box, without adaptation to the tasks. E.g., in Subtask 2, the entities are already given and need to be classified. We also identify the aspects.
5 Conclusion

We present a flexible, extensible open source system for document-level and aspect-based sentiment analysis and have reported state of the art results on two shared tasks in four different languages. Code and documentation are available on GitHub.\footnote{Code: https://github.com/uhh-lt/LT-ABSA} We also provide complete feature sets and trained models for all experiments reported in this paper.\footnote{Data and Models: http://ltdata1.informatik.uni-hamburg.de/sentiment/}

Acknowledgements

This work has been supported by the Innovation Alliance between the Deutsche Bahn and TU Darmstadt, Germany, as well as a DAAD WISE internship grant.

References


Naoaki Okazaki. 2007. CRFSuite: a fast implementation of Conditional Random Fields (CRFs).


Author Index

Becker, Christoph, 13
Biemann, Chris, 1, 55
Dakota, Daniel, 43
Daxenberger, Johannes, 22
Eger, Steffen, 22
Friedrich, Christoph M., 30
Gurevych, Iryna, 22
Hövelmann, Leonard, 30
Holschneider, Sarah, 1
Kübler, Sandra, 43
Kallmeyer, Laura, 18
Kumar, Abhishek, 55
Lanka, Soujanya, 36
Lee, Ji-Ung, 22
Mieskes, Margot, 13
Mishra, Pruthwik, 36
Mujadia, Vandan, 36
Naderalvojoud, Behzad, 18
Qasemizadeh, Behrang, 18
Ruppert, Eugen, 1, 55
Sayyed, Zeeshan Ali, 43
Schulz, Karen, 13
Sidarenka, Uladzimir, 49
Wojatzki, Michael, 1
Zesch, Torsten, 1