Aspect-Based Relational Sentiment Analysis Using a Stacked Neural Network Architecture

Soufian Jebbara and Philipp Cimiano

Semantic Computing Group, Cognitive Interaction Technology Center of Excellence (CITEC), Bielefeld University, Germany, {sjebbara,cimiano}@cit-ec.uni-bielefeld.de

Abstract. Sentiment analysis can be regarded as a relation extraction problem in which the sentiment of some opinion holder towards a certain aspect of a product, theme or event needs to be extracted. We present a novel neural architecture for sentiment analysis as a relation extraction problem that addresses this problem by dividing it into three subtasks: i) identification of aspect and opinion terms, ii) labeling of opinion terms with a sentiment, and iii) extraction of relations between opinion terms and aspect terms. For each subtask, we propose a neural network based component and combine all of them into a complete system for relational sentiment analysis.

The component for aspect and opinion term extraction is a hybrid architecture consisting of a recurrent neural network stacked on top of a convolutional neural network. This approach outperforms a standard convolutional deep neural architecture as well as a recurrent network architecture and performs competitively compared to other methods on two datasets of annotated customer reviews. To extract sentiments for individual opinion terms, we propose a recurrent architecture in combination with word distance features and achieve promising results, outperforming a majority baseline by 18% accuracy and providing the first results for the USAGE dataset. Our relation extraction component outperforms the current state-of-the-art in aspect-opinion relation extraction by 15% F-Measure.

1 Introduction

Sentiment analysis can be regarded as a relation extraction problem in which the sentiment of some opinion holder towards a certain aspect of a product, theme or event needs to be extracted. While most sentiment analysis methods extract an overall polarity score for a complete text, the following example clearly shows that this is not sufficient:

The **serrated portion** of the **blade** is **sharp**, but the **straight edge** was **marginal at best**.

The example shows an excerpt of a customer review regarding a kitchen knife. The opinion expression “sharp” constitutes a positive opinion towards the aspect “serrated portion”. In the same sentence, a negative opinion is expressed by the phrase “marginal at best” towards the aspect “straight edge”. Sentiment analysis needs to be regarded thus as a relation extraction problem consisting of three parts:

1. the extraction of **aspect** and **opinion terms** with respect to the discussed product, theme or event,
2. the labeling of these opinion terms with a **sentiment** (e.g. “positive”, “neutral”, “negative”), and
3. the extraction of **relations** between aspect and opinion terms.

In this work, we propose a complete and modular architecture that addresses all three tasks. The extraction of aspect and opinion terms can essentially be regarded as a tagging task and can potentially be tackled by sequence modeling techniques such as Hidden Markov Models, Conditional Random Fields (CRFs) etc. Besides, deep neural networks have received increasing interest in recent years and have been applied very successfully to a great variety of natural language processing (NLP)-related tasks. In particular, Convolutional Neural Networks (CNN) have been proposed as a general method for solving sequence tagging problems [5]. Also recurrent neural networks (RNN) have been applied to NLP-related tasks [2].

Building on these encouraging results, in this paper, we employ several neural network based components which we combine into a single architecture to address relational sentiment analysis in three steps. Firstly, we propose a component that combines convolutional neural networks with recurrent neural networks to extract aspect and opinion terms. Roughly, the method combines a CNN based sequence tagger with an RNN based tagger by stacking the RNN onto the CNN’s produced feature sequence. The motivation behind this approach is that the RNN on top of the CNN provides a way to preserve information over longer distances in the processed text. While the deep CNN based tagger is able to capture local dependencies around a word of interest, it cannot incorporate knowledge that appears far away from its current focal position. The RNN on top, however, might still be able to incorporate locally extracted features of the CNN from far preceding positions in a text through its recurrent hidden layer.

Secondly, a recurrent neural network extracts the expressed sentiment of each opinion term by using Part-of-Speech (POS) tags, word and distance embedding features. Our method considers the opinion terms in a wide context while still being able to label multiple opinion terms in a single sentence.

Thirdly, we extract aspect-opinion relations by using a similar RNN model to classify extracted aspect and opinion terms in a pairwise fashion. The combination of all these components allows us to realize sentiment analysis on a very fine-grained level, putting individual aspect and opinion mentions in relation. A schematic visualization of the complete architecture can be seen in Figure 1. Our contributions are the following:

- We present a complete architecture that addresses sentiment anal-
The paper is structured as follows: In the following section, we discuss related work from the domains of aspect-based sentiment analysis and relation extraction with respect to the individual subtasks these systems address. Next, in the sections 3, 4, and 5, we describe the components we use for our three subtasks. Section 6 describes our evaluation of the individual components that we apply to two datasets. We describe how parameters of the networks were optimized, the training procedure and the results for the different components on the two datasets. Lastly, we give a conclusion and discuss issues for future work.

1.1 Related Work

Our work is inspired by different related approaches. Overall, our work is in line with the growing interest of providing more fine-grained, aspect-based sentiment analysis [16, 13, 22], going beyond a mere text classification or regression problem that aims at predicting an overall sentiment for a text.

Vicente et al. [25] present a system that addresses opinion target extraction as a sequence labeling problem based on a perceptron algorithm with local features. The system also implements a sentiment polarity classifier to classify individual opinion targets. The approach uses a window of words around a given opinion target and classifies it with an SVM classifier based on a set of features such as word clusters, Part-of-Speech tags and polarity lexicons.

Aspect and opinion term extraction for sentiment analysis has also been addressed using probabilistic graphical models. Toh and Wang [32] for instance propose a Conditional Random Field (CRF) as a sequence labeler that includes a variety of features such as POS tags and dependencies, word clusters and WordNet taxonomies. Additionally, the authors employ a logistic regression classifier to address aspect term polarity classification. Klinger and Cimiano [12, 13] have modeled the task of joint aspect and opinion term extraction using probabilistic graphical models and rely on Markov Chain Monte Carlo methods for inference. They have demonstrated the impact of a joint architecture on the task with a strong impact on the extraction of aspect terms, but less so for the extraction of opinion terms. The approach to semantic role labeling proposed by Fonseca et al. [6] is closely related to our approach to aspect term extraction in that the task is phrased as a sequence tagging problem to which a convolutional neural network is applied.

Most relevant in terms of aspect and opinion term extraction are the works of Liu et al. [17] and Irsoy and Cardie [10]. Liu et al. address the extraction of opinion expressions while Irsoy and Cardie focus on the extraction of opinion targets. Both approaches frame the respective tasks as a sequence labeling task using RNNs.

Our work is related to other approaches using deep neural network architectures for sentiment analysis. Lakkaraju et al. [15] for example present a recursive neural network architecture that is capable of extracting multiple aspect categories and their respective sentiments jointly in one model or separately using two softmax classifiers.

Our approach to relation extraction is inspired by the work of Zeng et al. [34], who address a relation extraction task between pairs of entities using a convolutional neural network architecture. Their approach combines lexical features for both entities with sentence level features learned by a CNN model.

All of the above works address a subtask of relational sentiment analysis as we define it in this work, namely i) aspect and opinion term extraction ii) opinion-term specific sentiment extraction and iii) relation extraction. However, none of the above publications target all subtasks in a single architecture or offers the same degree of granularity in the sentiment analysis. We are thus the first to propose a

\[ \text{Figure 1. An architecture for sentiment analysis as a relation extraction problem.} \]

Here, we distinguish between the terminologies of aspect category extraction and aspect term extraction: The set of possible aspect categories is predefined and rather small (e.g. Price, Battery, Accessories, Display, Portability, Camera) while aspect terms can take many shapes (e.g. "sake menu", "wine selection" or "French Onion soup").
neural architecture that addresses all three subtasks within one system.

2 Datasets
For the evaluation of this work, we employ two datasets that provide annotated reviews for the task of relational sentiment analysis.

2.1 SemEval2015
The SemEval2015 Task 12 dataset [22] is used to evaluate our systems for the subtask of aspect term extraction. The dataset provides a collection of reviews from different domains (Restaurant, Laptops, Hotels), annotated on different aspect and sentiment levels. We only make use of the data from the restaurant domain as it contains annotations for explicitly mentioned aspect terms. The datasets for the laptop and hotel domains only contain annotations for aspect categories without annotated textual mentions. Note that we can only use this dataset to evaluate our architecture on the task of extracting aspect terms since the dataset is not annotated with respect to opinion terms, and therefore also without aspect-opinion relations.

2.2 USAGE
The USAGE corpus [14] is a collection of annotated English and German Amazon reviews of different product categories. The annotations include (among others) the mentioned aspect terms, the opinion terms marked with a sentiment and relations between aspect and opinion terms. We refer to Klinger and Cimiano [14] for a more detailed description of the dataset.

We restrict our use of this corpus to the annotations for the English reviews and follow the evaluation procedure based on 10-fold cross-validation and strict matching proposed by Klinger and Cimiano [14] for a more detailed description of the dataset.

We use the tag set provided by the Stanford POS tagger [33]. The tag set contains 45 tags plus one additional “padding” tag. We encode these tags in the following way:

<table>
<thead>
<tr>
<th>Tag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>Outside</td>
</tr>
<tr>
<td>I</td>
<td>Inside</td>
</tr>
<tr>
<td>O</td>
<td>Inside or Outside of an annotation</td>
</tr>
</tbody>
</table>

For most experiments, we instantiate two separate models – one for extracting aspect terms and another for extracting opinion terms – and train both separately to predict their respective tag sequences. However, as shown in Section 3.5 it is also possible to extract aspect and opinion terms jointly without using a larger tag set. For the actual sequence labeling, we compare convolutional neural networks, recurrent neural networks and combinations thereof since these are capable of dealing with sequential data.

In the following, we discuss the features used by our systems, which include both Part-Of-Speech tags as well as word embeddings. The word embeddings are learned from a domain-specific corpus of Amazon reviews using a skip-gram model [20]. Then, we present our different choices of components that we experimentally examine for the aspect and opinion term extraction subtask. Here, we describe a convolutional network architecture as well as a recurrent network architecture as baseline systems. We then describe a stacked architecture that feeds features of multiple convolutional layers to a recurrent layer that, in turn, produces a tag sequence. As a preview for future work, we also propose a component that extracts aspect and opinion terms jointly.

3 Aspect and Opinion Term Extraction
In this work, we compare different choices for neural network based components on the task of aspect and opinion term extraction. We interpret the extraction task as a sequence labeling task, similar to other sequence labeling tasks [32, 6] and predict sequences of tags for sequences of words. We use the IOB2 scheme [31] to represent our aspect and opinion annotations as a sequence of tags. According to this scheme, each word in our text receives one of 3 tags, namely L, O or B that indicate if the word is at the Beginning, Inside or Outside of an annotation:

\[
\text{The sake menu should not be overlooked!}
\]

\[
O \ B \ I \ O \ O \ O \ O \ O
\]

In this example the bold “sake menu” is an aspect term annotation that we encoded with the IOB2 scheme. We decided on encoding aspect term annotations separately from opinion term annotations thus resulting in two separate tag sequences per review. This procedure seems reasonable, since the USAGE dataset allows overlapping aspect and opinion term annotations. Encoding them into a single tag sequence would require to use a bigger tag set\(^2\) which we would expect to hinder the learning procedure.

\(^2\) The tag set would need to cover single and overlapping annotations using the following tag set: {I-Aspect, I-Opinion, I-Aspect-Opinion, B-Aspect, B-Opinion, B-Aspect-Opinion, O}

For most experiments, we instantiate two separate models – one for extracting aspect terms and another for extracting opinion terms – and train both separately to predict their respective tag sequences. However, as shown in Section 3.5 it is also possible to extract aspect and opinion terms jointly without using a larger tag set. For the actual sequence labeling, we compare convolutional neural networks, recurrent neural networks and combinations thereof since these are capable of dealing with sequential data.

In the following, we discuss the features used by our systems, which include both Part-Of-Speech tags as well as word embeddings. The word embeddings are learned from a domain-specific corpus of Amazon reviews using a skip-gram model [20]. Then, we present our different choices of components that we experimentally examine for the aspect and opinion term extraction subtask. Here, we describe a convolutional network architecture as well as a recurrent network architecture as baseline systems. We then describe a stacked architecture that feeds features of multiple convolutional layers to a recurrent layer that, in turn, produces a tag sequence. As a preview for future work, we also propose a component that extracts aspect and opinion terms jointly.

3.1 Features
Distributed word embeddings have proven to be a useful feature in many NLP tasks [5, 26, 16] in that they often encode semantically meaningful information about words [20, 26]. In this work, we employ word embeddings which we train on huge amounts of reviews since this data is closely related to our main application domain: relational sentiment analysis of customer reviews. The corpus includes the dataset of McAuley et al. [18, 19] which consists of \(\approx 83\) million reviews from 1996 to 2014. We train the skip-gram model [20] with hierarchical softmax as it is implemented in the topic modeling library gensim [24]. All reviews are lowercased and the dimensionality of the vector of word embeddings is set to \(D_{\text{word}} = 100\). Rare words that appear less than 10 times in the corpus are replaced with a special token \(<UNK>\). This token is later used to represent previously unseen words in order to provide a vector for each word at test time. The resulting vocabulary contains \(\approx 1\) million unique words, which we trim to the 200000 most frequent words.

To quantify the impact of using these domain-specific embeddings, we also compute word embeddings on a domain-independent corpus of Wikipedia articles. As we show in Section 6.2, the domain-specific word embeddings indeed outperform the more general Wikipedia word embeddings.

Additionally to the sequence of word embeddings, we evaluate the use of corresponding Part-Of-Speech tags for each word that we obtain from the Stanford POS tagger [33]. The tag set contains 45 tags plus one additional “padding” tag. We encode these tags in the One-Hot\(^3\) encoding, which results in a POS tag feature vector with \(D_{\text{pos}} = 46\) dimensions for each word.

3.2 Convolutional Neural Network Model
While convolutional neural networks were originally intended for image processing, they have been successfully applied to several natural language processing tasks as well [23, 11, 27]. When working with sequences of words, convolutions allow to extract local features around each word.

The CNN component for sequence tagging that we propose is composed of several sequentially applied layers that transform an

\(^3\) A vector of 0s with a single 1 to represent the specific tag.
When using POS tags as additional features, we concatenate the corresponding tags. This sequence of tags encodes predicted aspect and opinion annotations for the given review. More formally, the process from word sequence to tag sequence can be described as follows:

Given a sequence of arbitrary length $N$ of words:

$$[w]^N = \{w_1, \ldots, w_N\}$$

that correspond to a vocabulary $V$, our model applies a word embedding layer to each sequence element to retrieve a sequence of word embeddings $u_n \in \mathbb{R}^{D_{\text{word}}}$:

$$[u]^N = \{u_1, \ldots, u_N\}.$$ 

This is done by treating the embedding matrix $W_{\text{word}} \in \mathbb{R}^{D_{\text{word}} \times |V|}$ as a lookup table and returning the column vector that corresponds to the respective word index.

Then, each window of $l_{\text{conv}}$ consecutive vectors in $[u]^N$ around $u_n$ is convolved into a single vector $h_{\text{conv}}^1 \in \mathbb{R}^{D_{\text{conv}}}$, where $D_{\text{conv}}$ specifies the number of feature maps for this convolution. Precisely, the convolution is performed on the concatenated vector $z_n \in \mathbb{R}^{D_{\text{word}} \times l_{\text{conv}}}$ that we define as:

$$z_n = u_n - (l_{\text{conv}} - 1)/2 \oplus \ldots \oplus u_{n + (l_{\text{conv}} - 1)/2},$$

where $\oplus$ represents the concatenation operation. The convolution at position $n$ in the sequence is then:

$$h_n^1 = \sigma(W_{\text{conv}}z_n + b_{\text{conv}}),$$

where the kernel matrix $W_{\text{conv}} \in \mathbb{R}^{D_{\text{conv}} \times D_{\text{word}} \times l_{\text{conv}}}$ and the bias vector $b_{\text{conv}}$ are shared across all windows for this convolution. The function $\sigma$ is an element-wise non-linear activation function such as the rectified linear function $f(x) = \max(0, x)$. The resulting sequence is then:

$$[h^1]^N = \{h_1^1, \ldots, h_N^1\}.$$ 

This convolution operation can be applied several times (with different $W_{\text{conv}}, b_{\text{conv}}, l_{\text{conv}}$, and on the output sequence of the previous convolution) to yield a sequence of more abstract representations:

$$[h^m]^N = \{h_1^m, \ldots, h_N^m\}.$$ 

In a last step we apply a standard affine neural network layer with a softmax activation function to each individual sequence element that projects the hidden representation $h_n^m$ to a vector of $D_{\text{IOB2}} = 3$ probability scores that represent the word’s affiliation to the corresponding tags I, O, or B.

In our following experiments for aspect and opinion term extraction, this CNN architecture consists of a word embedding lookup table for $D_{\text{word}} = 100$ dimensional vectors, 3 convolution layers and a dense layer that is applied to each single sequence element. We chose a kernel size $l_{\text{conv}} = 3$ and $D_{\text{conv}} = 50$ feature maps with a rectified linear activation function. We do not employ a max pooling operation after convolutions in order to retain the initial sequence length. However, we employ dropout with a drop probability of 0.5 after each convolution as a regularization to prevent overfitting. The final dense layer uses a softmax activation function to compute probabilities for each of the 3 tags (I, O or B). The hidden layer sizes for this model are therefore 100-50-50-50 (including the embedding layer). Figure 2 depicts the architecture of this network.

---

4 When using POS tags as additional features, we concatenate the corresponding one-hot vector $p_t$ to the word embedding $u_n$ and use the resulting sequence $[w]^N = \{(u_1 \oplus p_t), \ldots, (u_N \oplus p_t)\}$ in the next steps.

5 Since we want to apply the convolution operation to the first and the last element in a sequence, too, we pad the input sequence with vectors of 0s.
of high-level features, we stack a GRU layer that learns temporal dependencies of its input sequence. Again, a dense layer with a softmax activation is used to map the recurrent hidden states to tag probabilities. The stacked CNN-RNN model uses the hidden layer sizes 100-50-50-50-100 (or 146-50-50-50-100 when using additional POS tag features).

### 3.5 Joint Model

Klinger and Cimiano [12, 13] present models that are able to extract aspect and opinion terms jointly, leveraging knowledge about aspect terms in order to find opinion terms and vice versa. Using the stacked model as a basis, we try to replicate this behavior and construct a model that can infer aspects and opinions jointly.

This model has a very similar structure as the stacked architecture from the previous section. It differs, however, in a second output layer that we connect to the GRU layer. With that, the model is able to predict two tag sequences at once: one for extracted aspect terms, the other for opinion terms. See Figure 3 for a depiction of the joint architecture.

```
<table>
<thead>
<tr>
<th>Input and Embedding Layer</th>
<th>Convolution and GRU Layers</th>
<th>Output Layers</th>
</tr>
</thead>
<tbody>
<tr>
<td>the</td>
<td></td>
<td></td>
</tr>
<tr>
<td>tuna</td>
<td></td>
<td></td>
</tr>
<tr>
<td>wasabe</td>
<td></td>
<td></td>
</tr>
<tr>
<td>potatoes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>are</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

Figure 3. The CNN-RNN network for joint aspect and opinion term extraction. Solid boxes at the input mark aspect terms and are tagged with the correct tags at the aspect output layer (solid arrows and boxes). The dashed box at the input marks an opinion term. The corresponding output layer is marked with dashed arrows and boxes.

### 4 Opinion Term Specific Sentiment Extraction

With the previously described models, we are able to detect mentioned aspect and opinion terms, however, without extracting the individual sentiment. The second step in our pipeline for relational sentiment extraction is to extract sentiments for these opinion terms. The difficulty here is that it is not enough to extract an overall sentiment for a given review. Rather, the sentiment needs to be extracted with respect to one of possibly several opinion terms in a text. In this section, we propose a recurrent neural network architecture together with a combination of word and distance embedding features to address this task.

Given an already detected opinion term in a review, we tag each word in the review with its relative distance to the opinion term in question, as shown in the following example:

<table>
<thead>
<tr>
<th>Coffee stays fresh and hot in the Carafe</th>
<th>(Text)</th>
</tr>
</thead>
<tbody>
<tr>
<td>NN</td>
<td>VBZ</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

where the bold words “stays fresh” form the opinion term for which we want to extract the sentiment. The italic word “hot” is another annotated opinion term that is neglected in this extraction step. Below, the sequence of corresponding POS tags (as obtained with the Stanford POS-tagger) and the relative word distances (O) to the opinion term are shown.

We extract a window of \( l_{\text{pol}} = 20 \) words centered around the opinion term instead of considering the whole review text. Sequences for review texts with less than \( l_{\text{pol}} \) words are padded at the left with Os. Analogously, the sequence of POS tags is extracted. Using a subsequence of words and POS tags around each opinion term is reasonable since the lengths of the reviews in the USAGE corpus reach up to several hundred words. Taking the whole review text into account for all opinion terms is computationally very demanding.

We convert each word into its respective vector representation using the pretrained lookup table of word embeddings, thus obtaining a sequence of word vectors. Similar to this, we also use an embedding layer for the relative distances that provides us with a learned vector representation of dimensionality \( D_{\text{dist}} = 10 \) for each distance value, similar to the approach proposed by Zeng et al. [34]. While we did not evaluate those distance embeddings extensively, first results suggest that there is indeed a benefit in mapping the distances to real valued vectors instead of using the raw distance values. The benefits of distance and position embeddings are supported by the results of other works [21, 34, 29].

The individual vectors of the three sequences – word embeddings \( u_n \), POS tags \( p_n \) and distance embeddings \( d_n \) – are concatenated, resulting in a single sequence of length \( l_{\text{pol}} \) with \( D_{\text{word}} + D_{\text{pos}} + D_{\text{dist}} \) dimensional elements. We feed the resulting sequence to a recurrent neural network consisting of three layers. The first hidden layer is a GRU layer with \( D_{\text{gru}} = 100 \) hidden units that reads in the sequence of vectors and produces a sequence of hidden states. The second layer is a densely connected layer of maxout units [7] that transforms the final hidden state of the GRU layer into a vector \( h' \) of \( D_{\text{pol}} = 100 \) dimensions. Lastly, another maxout layer maps the previous hidden layer to \( 4 \) output units using a softmax activation function. Each of the \( 4 \) output units corresponds to one of four possible sentiments: positive, neutral, negative and unknown. The sentiment with the highest probability at the corresponding output unit is the predicted sentiment. Figure 4 visualizes the network.

```
<table>
<thead>
<tr>
<th>Input and Embedding Layers</th>
<th>GRU Layer</th>
<th>Maxout Layer</th>
<th>Output Layer</th>
</tr>
</thead>
<tbody>
<tr>
<td>the</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>tuna</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>wasabe</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>potatoes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>are</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

Figure 4. The component for opinion term specific sentiment extraction. The dashed box at the input marks an opinion term. Besides, the corresponding POS tags and the relative word distances are shown. Here, the input vectors are composed of three parts: one vector for the word embeddings, one vector for the POS tag encoding and one vector for the distance embedding. The output layer contains one unit for each possible sentiment label: positive, neutral, negative and unknown.
5 Aspect-Opinion Relation Extraction

This section introduces the component that is responsible for extracting relations between already extracted aspect and opinion terms.

Besides the annotations for aspect and opinion terms, the USAGE corpus provides annotations of relations between pairs of these aspects and opinions, which we simply refer to as aspect-opinion relations\(^6\). The presence of such a relation implies that the opinion term targets the annotated aspect term. Due to this binary nature of the relations (present or not-present) in the USAGE corpus, our relation extraction approach predicts a boolean tag for any given pair of aspect term and opinion phrase. This pairwise approach allows us to predict the many-to-many relations between aspect and opinion term that are present in the corpus.

Again, we employ a neural network based model that is very similar in structure to the component for opinion term specific sentiment extraction (see previous section). We adopt a similar strategy as presented in Zeng et al. \[34\] to address relation extraction. In contrast to Zeng et al. \[34\] however, we use a recurrent neural network instead of a convolutional architecture to perform the actual relation extraction.

Our approach employs four types of features for a given pair of aspect and opinion term:

- the sequence of word embeddings of length \(l_{\text{rel}} = 20\) that is centered around the aspect and the opinion,
- the sequence of corresponding POS tags for each word,
- a sequence of relative distances of each word to the aspect term, and
- a sequence of relative distances of each word to the opinion term.

As a first step, our model computes the distance (in words) between the two terms and automatically rejects all pairs which are more than 20 words apart from each other. While this does reject some valid relations, we can still predict 98% of relations correctly for this maximum distance of 20 words. For those pairs which are below the maximum distance, our model extracts a subsequence of word embeddings of length 20, centered around the aspect term and opinion term. Sequences for review texts with less than 20 tokens are padded at the left with 0s. Analogously, the sequence of POS tags is extracted.

To encode information about the distance between the aspect and opinion phrases with respect to their position in the review, we follow a similar approach as Zeng et al. \[34\] and Nogueira dos Santos et al. \[21\]. We guide the network’s attention to the targeted aspect and opinion terms by computing the relative distances of each word in the sequence to the aspect term and to the opinion term, respectively.

<table>
<thead>
<tr>
<th>Input and Embedding</th>
<th>GRU Layer</th>
<th>Maxout Layer</th>
<th>Output Layer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Embedding Layers</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>the DT -1 -6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>and CC 1 -4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>wasabi NN 2 -3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>potatoes NN 3 -2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>are VBP 4 -1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>excellent JJ 5 0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>. . . . 6 1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 5. The component for aspect-opinion relation extraction. The solid and dashed boxes at the input mark aspect and opinion terms, respectively. Besides, corresponding POS tags and the relative word distances to the aspect and opinion term are shown. The input vectors are split into four parts: word embeddings, POS tag encoding, aspect distance embeddings and opinion distance embeddings. The output layer contains one single sigmoid unit. An output value \(>0.5\) signifies a relation between the aspect and opinion terms.

6 Experiments and Results

This section evaluates our proposed architecture for relational sentiment analysis. Our evaluation targets the individual components of the overall system, measuring their performances in isolation. For the evaluation, we consider the two datasets described in Section 2. Both datasets offer a different granularity in their annotations. The SemEval dataset does not provide annotations for opinion terms, opinion term specific sentiment or aspect-opinion relations. As such, we only evaluate our component for aspect term extraction on this dataset. The USAGE dataset, on the other hand, offers annotations for all our subtasks, allowing us to evaluate all our components on this dataset. Where possible, we give precision, recall and \(F_1\) scores for our own and baseline approaches.

All experiments were performed with the deep learning library Keras \[3\] and employ many of its pre-implemented algorithms. Tag sequences predicted by our approach are post-processed to yield only sequences that are valid according to the IOB2 scheme.

\(^6\) The creators of the USAGE corpus actually refer to these relations as TARGET-SUBJ. To keep a consistent terminology in this work, we call them aspect-opinion relation.
6.1 Training
This section briefly outlines our training procedure. Since we deal with sequences of variable lengths, training our models in mini batches would require to pad shorter sequences with special padding elements. Due to the large differences in the length of reviews in the USAGE corpus (18 to > 2000 words), we avoid padding sequences with too many padding elements. Instead, we train our models on one data sample at a time. The optimization of the models’ weights is performed with RMSProp [30], which converges after a few epochs.

Since the performance did not depend strongly on the number of iterations over the training data, we trained all aspect and opinion term extraction models for 15 epochs. The component for sentiment extraction was trained for 14 epochs and the relation extraction component performed best with 28 epochs of training.

6.2 Initialization of Word Embeddings
In our first experiment, we compare the performance impact of initializing the weights of the word embedding lookup table in the stacked CNN-RNN model randomly to the initialization with our pretrained embeddings. Our intuition is that both sets of pretrained embeddings are helpful for the extraction of aspect and opinion terms but more so the domain-specific embeddings. These are expected to capture the most task relevant semantics which might help with the extraction task.

Since many of our network parameters are initialized randomly and our training data is processed in a random order, the performance of our model might be influenced by these external factors. To mitigate these effects, we performed each experiment three times and averaged the outcomes.

The experiments for aspect term extraction on the SemEval2015 dataset show that our model achieves on average an \( F_1 \) score of 0.581 with the randomly initialized embeddings. The initialization with the Wikipedia embeddings leads to a much higher score, namely \( F_1 = 0.618 \), while the domain-specific review embeddings yield an F-Measure of 0.646. As expected, we can observe a large benefit in pretraining our word embeddings on huge collections of natural language texts with the largest gain using domain-specific embeddings.

Considering these first results, we only use the domain-specific review embeddings as initialization in further experiments.

6.3 Evaluation: Aspect and Opinion Term Extraction
This section evaluates our relational sentiment analysis architecture focusing on the component for aspect and opinion term extraction. We perform the evaluation in two steps: First, we evaluate the component on the SemEval dataset measuring its performance for aspect term extraction only. Keep in mind that we can neither evaluate opinion term extraction nor opinion term specific sentiment analysis or aspect-opinion relation extraction on this particular dataset due to granularity of the provided annotation. Those subtasks, however, are evaluated in Sections 6.3.2, 6.4 and 6.5.

6.3.1 CNN vs. RNN vs. Stacked Models
This section evaluates our aspect term extraction component and considers different neural models. We train and test the CNN, the RNN and the stacked CNN-RNN model from the sections 3.2, 3.3 and 3.4 on the SemEval2015 dataset using the official training and test split.

Again, we perform each experiment three times to alleviate differences due to the networks’ initializations and training sample orders. Table 1 shows the average precision, recall and \( F_1 \) score for each model.

<table>
<thead>
<tr>
<th>Model</th>
<th>( P )</th>
<th>( R )</th>
<th>( F_1 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>RNN</td>
<td>0.592</td>
<td>0.646</td>
<td>0.618</td>
</tr>
<tr>
<td>CNN</td>
<td>0.558</td>
<td>0.702</td>
<td>0.621</td>
</tr>
<tr>
<td>Stack</td>
<td>0.599</td>
<td>0.703</td>
<td>0.646</td>
</tr>
<tr>
<td>Stack+POS</td>
<td>0.633</td>
<td>0.689</td>
<td>0.659</td>
</tr>
<tr>
<td>EliXa</td>
<td>–</td>
<td>–</td>
<td>0.701</td>
</tr>
</tbody>
</table>

Table 1. Results for aspect term extraction on the SemEval2015 test dataset using different model architectures and additional POS tag features. EliXa represents the current state-of-the-art.

The model based on convolutional layers and the model based on recurrent layers both perform similarly regarding the overall \( F_1 \) score. Combining both types of models in a stack-like architecture results in an increased averaged \( F_1 \) score that is statistically significant with \( p = 0.05 \). Providing additional features in the form of POS tags does also improve the model’s performance. While we still perform not quite as good as the current state-of-the-art system EliXa [25] we do perform well with respect to the overall ranking of the SemEval2015 task as can be seen in Pontiki et al. [22].

In spite of being a few percentage points below the current state-of-the-art system EliXa, our proposed method still constitutes a meaningful contribution. The benefit of our method is that it is not restricted to the mere extraction of aspect terms but that it is capable of extracting aspect and opinion terms jointly. The following section shows that our method achieves state-of-the-art performance on the USAGE dataset with this more complex joint extraction.

Based on the results of these experiments, we perform all following aspect and opinion phrase extraction tasks using the CNN-RNN model with additional POS tag features.

6.3.2 Joint vs. Separate Models
This section investigates the benefits of predicting aspect and opinion phrases of the USAGE corpus jointly in one model, in contrast to two separate models. For this, we consider two joint models with different hidden layer sizes, namely with 100-50-50-50-100 (dubbed Joint small) and 100-100-100-100-200 (dubbed Joint large). This should account for differences in performance that solely result from different network capacities. As proposed by Klinger and Cimiano [14], we evaluate our models by 10-fold cross validation. Table 2 shows the results for the joint and the separate models.

We can see that extracting aspect and opinion terms jointly does indeed enhance the model’s performance, but only so for a larger network configuration. The extraction of opinion terms benefits from the joint setting in particular. However, this might also be attributed to the increased size of the hidden layers in our neural architecture. The extraction of opinion terms might simply require more network parameters which it is able to claim in the larger joint architecture. A more detailed investigation needs to be conducted in future work.

Bear in mind that we do not compare our results on this dataset with the EliXa system referenced in Section 6.3.1. The EliXa system is applicable to aspect term extraction in isolation and is not designed for opinion term extraction.
Table 2. Performances for aspect and opinion phrase extraction on the USAGE dataset for joint and separate models. Joint small and Joint large are the joint models with hidden layer size 100-50-50-50-100 and 100-100-100-100-200, respectively.

<table>
<thead>
<tr>
<th>Model</th>
<th>P</th>
<th>R</th>
<th>F1</th>
<th>P</th>
<th>R</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aspect only</td>
<td>0.63</td>
<td>0.70</td>
<td>0.66</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Opinion only</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>0.44</td>
<td>0.48</td>
<td>0.45</td>
</tr>
<tr>
<td>Joint small</td>
<td>0.57</td>
<td>0.65</td>
<td>0.61</td>
<td>0.40</td>
<td>0.40</td>
<td>0.40</td>
</tr>
<tr>
<td>Joint large</td>
<td>0.65</td>
<td>0.69</td>
<td>0.67</td>
<td>0.47</td>
<td>0.53</td>
<td>0.50</td>
</tr>
</tbody>
</table>

Table 3. Accuracy for opinion term specific sentiment extraction on gold annotations.

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
<th>#Correct</th>
<th>#Incorrect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive Only</td>
<td>0.647</td>
<td>342.6</td>
<td>189.5</td>
</tr>
<tr>
<td>Our Approach</td>
<td>0.831</td>
<td>441.6</td>
<td>90.5</td>
</tr>
</tbody>
</table>

6.4 Evaluation: Opinion Term Specific Sentiment Extraction

Next, we show our evaluation of the component proposed in Section 4. The model considers individual opinion terms in a wide context and predicts one of four sentiment labels for each presented opinion term.

We perform the sentiment extraction on the opinion terms of the gold standard annotations of the USAGE corpus in order to measure the sentiment extraction in isolation. To keep the experiments consistent across our different components, we perform the prediction on a 10-fold cross validation of the USAGE documents. The results in Table 3 show the average accuracy7 of predicting the sentiment label of an opinion term with the number of correctly and incorrectly labeled opinion terms. We also show the results of a naive approach that always predicts the (most frequent) sentiment label positive to act as a simple baseline.

Table 4. F1 score for aspect-opinion relation extraction on gold annotations.

<table>
<thead>
<tr>
<th>Model</th>
<th>P</th>
<th>R</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Klinger2014</td>
<td>0.75</td>
<td>0.81</td>
<td>0.81</td>
</tr>
<tr>
<td>Our Approach</td>
<td>0.87</td>
<td>0.85</td>
<td>0.85</td>
</tr>
</tbody>
</table>

7 We report accuracy since precision, recall and F1 score are all equal in the case where the number of annotations is fixed.

6.5 Evaluation: Aspect-Opinion Relation Extraction

This part of our evaluation focuses on the last component in the overall architecture for relational sentiment analysis that is responsible for the extraction of aspect-opinion relations. As before, we perform the relation extraction on the aspect and opinion terms from the gold standard annotations of the USAGE corpus, in order to measure our system’s performance for relation extraction in isolation. This methodology is adopted from Klinger and Cimiano [14] and allows us to compare our method to their work. Table 4 shows the results of a 10-fold cross-validation of our proposed component. The results show that our RNN-based model improves relation extraction by 15% F-Measure compared to the probabilistic graphical model of proposed by Klinger and Cimiano [14].

7 Conclusion and Future Work

In this work, we presented a modular architecture that addresses sentiment analysis as a relation extraction problem. The proposed architecture divides the problem into three subtasks and addresses each with a dedicated component. This highly flexible approach offers a fine-grained solution for sentiment analysis.

As part of this overall architecture, we presented possible implementations for the individual components: First, we presented different neural network models that are capable of aspect and opinion term extraction and which achieved competitive and state-of-the-art results on different datasets. We could report a benefit for this task in using domain-specific word embeddings compared to domain-independent and randomly initialized embeddings. We investigated the extraction of aspect and opinion terms separately and jointly and found the joint approach to produce superior results in one setting. Thus, we confirm previous results from Klinger et al. [12] who used a probabilistic graphical model instead of a neural network model.

Secondly, we addressed opinion term specific sentiment extraction with a recurrent neural network model and distance embedding features and achieved promising results; the first sentiment extraction results on the considered dataset.

Thirdly, as another contribution, we designed and evaluated a model for the extraction of relations between aspect and opinion terms which outperformed prior results on the same dataset. Our work shows that it is possible to divide sentiment analysis in a flexible and fine-grained way using a highly modular architecture.

All proposed components stand out by their minimal use of hand-engineered features that are strongly tuned to their specific tasks. The only external resources that were used are machine-generated POS tags and word embeddings which were created with a data-driven approach. Nevertheless, all components perform competitive on their individual subtasks. It is easily conceivable to provide further task-specific features to improve the performances of the individual components even further. This, however, is not the goal of this work, which is why we leave this part for future work.

Furthermore, we expect our components to benefit from bidirectional recurrent connections [28] so that words appearing later in a sentence can be taken into account. The investigation of attention mechanisms for RNNs [1] seems also very promising to allow the models to focus more strongly on important parts of the input sequence.

ACKNOWLEDGEMENTS

This work was supported by the Cluster of Excellence Cognitive Interaction Technology ’CITEC’ (EXC 277) at Bielefeld University, which is funded by the German Research Foundation (DFG).
REFERENCES


