Comparing Sentiment Analysis Classification of Discourse Units and Sentences

Tim Pfeifle, Li Canchen, Hendrick Pauthner
Technical University Munich - Lab Course Opinion Mining

July 30, 2019

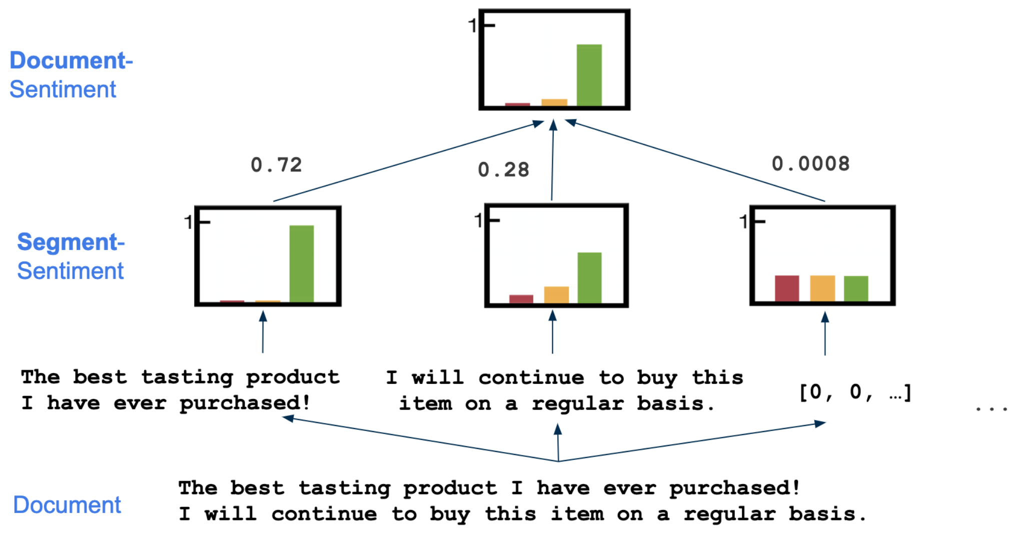
1 Introduction

Online blogs, social media sites, and online product reviews present us with countless opinions of other people. Many users rely on those opinions when researching products they want to buy. John Horrigan stated that 81% of Internet users have actually used online reviews for exactly this purpose (Horrigan, 2008). But while more and more reviews are available, many of them do not contain explicitly labeled sentiment, e.g. in the form of a star rating. As users can not be expected to manually read through all the available reviews, extracting the sentiment of those reviews automatically is an increasingly important task.

Sentiment analysis consequently uses this textual data to computationally study people‘s opinions towards entities, such as products or political parties. The applications of this technique are very diverse. With sentiment analysis, companies, e.g., can evaluate their customer’s attitude towards their products and governments can dig the public‘s attitude toward specific policies.

The most common approach for sentiment analysis is to regard the documents as atomic entities and predict their sentiment as a whole, even though documents often convey a mixture of positive and negative sentiment. Learning to separate those sub-document sentiments is especially important for the task of opinion summarization. This gave rise to the idea of splitting documents into fine-grained segments and using multiple instance learning (MIL) to classify each segment independently. Those separate sentiments are then combined into a document sentiment (see Figure 1).

To obtain the segments of a document, one can either segment the document by sentences, or by elementary discourse units (EDU), according to the rhetorical structure theory (RST). In our work, we compare different segmentation strategies for multiple instance learning and their performance when using different text embeddings. Unfortunately, most data sets only contain document and not segment level annotations as a ground truth. Creating data sets with the fine-grained segment level annotations is consequently an important task to evaluate and increase the performance of MIL. Therefore, we provided a segment-wise annotation task for the organic data set.



Document-Sentiment
1
0.72
0.28
0.0008
Segment-Sentiment
1
1
1
The best tasting product I have ever purchased!
I will continue to buy this item on a regular basis.
[0, 0, …]
...
Document
The best tasting product I have ever purchased! I will continue to buy this item on a regular basis.

Figure 1: Multiple instance learning for sentiment analysis

2 Related Work

The primary structure of our model follows MILNET (Angelidis and Lapata, 2018), which predicts the sentiment of segments in a document and then combines those predictions into a document-level sentiment.

Segmentation While the most common form to segment a document is by sentence, alternatives include segmentation by EDUs or phrase level (Socher et al., 2013). In our work, we focused on the comparison between EDUs and sentences.

To parse the document into EDUs there exist many Rhetorical Structure Theory style parsers. They do not only generate the EDU segmentation for the document but also create a full Rhetorical Syntax Tree containing the relationships between these EDUs. An extensive comparison of different RST parsers was done in (Morey, Muller and Asher, 2017), according to which the performance of the best RST parser is already quite close to the performance of human annotation. We adopted the work from Feng and Hirst (Feng and Hirst, 2014) as they achieved the highest accuracy for EDU segmentation (Morey, Muller and Asher, 2017) and have made their parser publicly available.[1]

Multiple Instance-Learning In multiple instance learning (MIL) labels are assigned to a group of instances (document), while the labels of the individual instances are unknown. E.g., the sentiment of a product review is known, but the sentiment of the individual sentences is not known. This situation is common, as many data sets only provide the document sentiment.

HIERNET (Yang et al., 2016) uses MIL to produce segment representations which are combined by attention weights into a document sentiment prediction. The main difference to the MILNET (Angelidis and Lapata, 2018) approach we are following is, that HIERNET does not classify the segments, but only produces an encoding for them. Both models do only take the document level annotations as input.

[1]http://www.cs.toronto.edu/ weifeng/software.html

Embedding As illustrated in (Kim, 2014), convolutional neural networks (CNNs) with different kernel heights perform well for encoding text in classification problems and are used in many state-of-the-art sentiment classification models. Our model will adopt this approach as well. While one can train embeddings using CNNs, using pre-trained text embedding models such as word2vec (Mikolov et al., 2013), fastText (Bojanowski et al., 2017), and XLING (Rücklé et al., 2018) can increase the performance of text classification problems, including sentiment classification. Consequently, we will compare the different embeddings.

3 Methodology

Our model is based on the assumption that each segment contains a different degree of sentiment. The segment-wise sentiment might even contradict other segments in the same document. Therefore, we predict the sentiment for each segment independently and then combine these predictions into one document-wise sentiment prediction by weighting each segment with its importance (see Figure 1). This importance we learn using an attention mechanism.

3.1 Document Segmentation

We compared two different approaches for splitting the documents into segments (see Figure 2).

Sentences

The most intuitive way is to split the document at sentence delimiters, such as '.', '!', and '?', into a set of sentences. This represents the assumption that sentences separate the different sentiments in one review.

Discourse Units

An alternative approach uses the introduced Rhetorical Structure Theory (RST) parser (Feng and Hirst, 2014) to parse the document into sub-sentence units, so-called *elementary discourse units* (EDUs). The parser extracts independent clauses in sentences and thereby creates a tree representation of the document. As the EDUs roughly represent independent segments, Stefanos Angelidis et. al. (Angelidis and Lapata, 2018) proposed to use them as document segments. Those units are more fine-grained than the sentence-wise segments and one of our goals was to determine whether or not they contain sub-sentence sentiment or not (see Figure 6. Following MILNET we only use the EDUs and not the tree representation.

3.2 Segment Encoding

As in other opinion mining tasks we have to map the words to vectors of real values, to use them as input for our model. While there are many different options to choose from, for such a so-called embedding, we grouped them into the following categories:

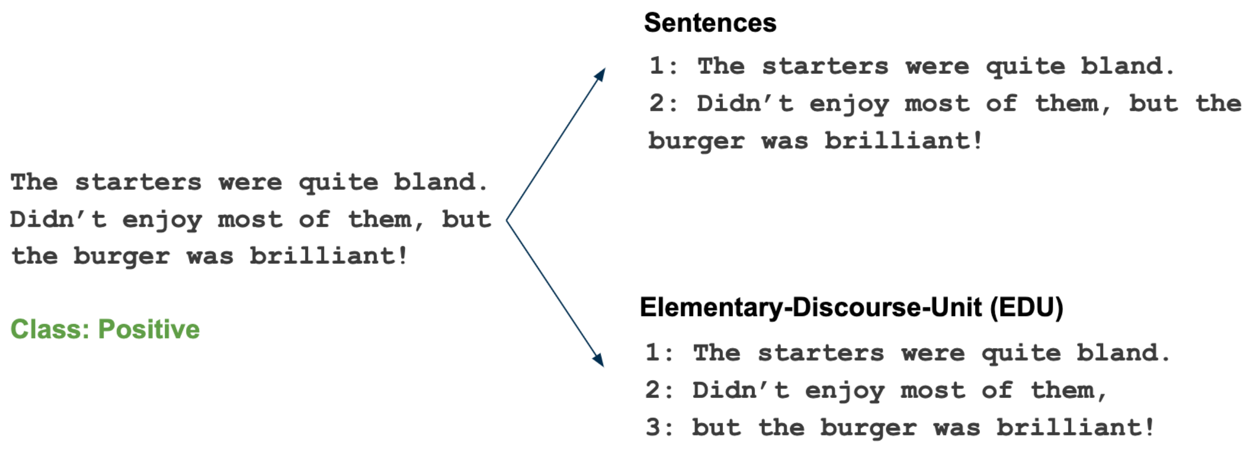

The starters were quite bland. Didn't enjoy most of them, but the burger was brilliant!
Class: Positive
Sentences
1: The starters were quite bland.
2: Didn't enjoy most of them, but the burger was brilliant!
Elementary-Discourse-Unit (EDU)
1: The starters were quite bland.
2: Didn't enjoy most of them,
3: but the burger was brilliant!

Figure 2: Alternative options for document segmentation

Segment Level Embedding

Those embeddings create a real-valued vector, not for each word, but instead directly for a complete segment, such as a sentence or an EDU. The Universal Sentence Encoder Cross-lingual (XLING) provides segment level encoding and is trained on multiple languages, including but not limited to English, German, and Chinese. In our work, we use XLING as the segment level encoding.

Word Level Embedding

Alternatively one can also create an embedding for each word separately. Therefore, a segment will be encoded not by one vector, but instead by a list of vectors. Two such embeddings that are very common are the word2vec and the fastText embedding, both of which generate a 300-dimensional vector for each word.
Those word embeddings are then concatenated into a segment matrix and then combined into a segment embedding with a multi-kernel 1-dimensional CNN network (Angelidis and Lapata, 2018) as seen on the left in Figure 3. By using multiple kernel heights h we compute multiple feature vectors $c = [c_1, c_2, \dots, c_{n-h+1}]$ which we aggregate into the final segment encoding using max-pooling.

3.3 Predicting Sentiment

For each segment, we predict the sentiment independently using the same sentiment classifier. Note that we do not have any annotations about the sentiment on the segment level, but only on the document level. Therefore, our loss is based on the document level prediction and only back-propagated through this segment level prediction. We then combine the independent segment level predictions using attention, to weight the importance of each segment, into the document level prediction.

Classification Model

The predictions p_i for each segment embedding v_i are learned with a simple linear layer and then normalized with the softmax function (Eq. 1). Since our model is a multiple instance learning model,

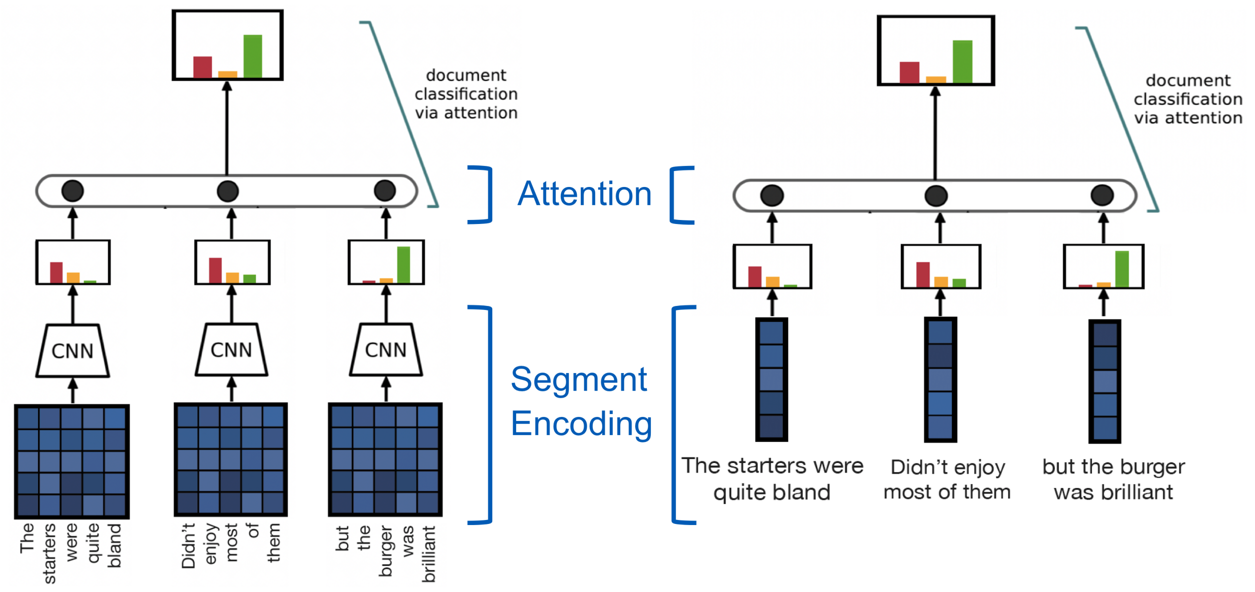

document classification via attention
Attention
Segment Encoding
CNN
The starters were quite bland
Didn't enjoy most of them
but the burger was brilliant

Figure 3: **Left:** Using word level embedding **Right:** Using segment level embedding

the weights of this layer are shared for all segments. p_i represents the class probability distribution for the i-th segment. Those distributions correspond to the bar-charts in Figure 3.

$$p_i = softmax(W_c v_i + b_c) \tag{1}$$

Attention Model

To focus on the segments that convey more sentiment than others we use an attention layer. The attention weights are generated from the segment encodings v_i. We first use two separate GRU models to create a hidden vector for each segment's encoding vector v_i (Eq. 2).

$$h_i = [\overrightarrow{GRU(v_i)}, \overleftarrow{GRU(v_i)}], i \in [1, m] \tag{2}$$

The hidden vector h_i is then passed through a dense layer with a $tanh$ activation function (Eq. 3).

$$h_i^{'} = tanh(W_a h_i + b_a) \tag{3}$$

We train the key h_a to recognize sentiment-heavy segments. From this, we determine the attention weights (see Eq. 4) and scale them with a softmax function so that they represent a probability distribution.

$$a_i = \frac{exp(h_i^{'\top} h_a)}{\sum_i exp(h_i^{'\top} h_a)} \tag{4}$$

Finally, the attention weights are used to weight the prediction for each segment and give the final class distribution for the review document p_d (Eq. 5). This p_d corresponds to the histogram at the top of Figure 1.

$$p_d^{(c)} = \sum_i a_i p_i^{(c)}, c \in [1, C] \tag{5}$$

Segments with less sentiment should get a small attention-weight a_i, while more important segments should get a large a_i and therefore have more impact on the document sentiment. For this document-wise sentiment p_d our data sets contain ground truth data so we use the negative log-likelihood of its prediction as our loss.

4 Data

We trained our model on two subsets of the Amazon product data (He and McAuley, 2016).

The organic data set unfortunately does not provide document-wise annotations. While it is possible to feed only a single sentence as input into the model we are not able to perform multiple instance learning then. Therefore, we decided to run our model only on the Amazon product data and provide the annotation task on the organic data set.[2] An overview of the different data sets is given in Table 1. We split each data set into training (80%), validation (10%) and testing (10%).

	Electronics [3]	Food [4]	Organic [5]
Documents	245,799	290,463	5561
Average #Sentences	5.4	4.1	1
Average #EDUs	8.2	9.6	3.2
Average #Words	76.3	68.2	20.7
Vocabulary Size	197,609	165,018	2728
Classes	5	5	3

[1] Amazon product data, Category: "Electronics"
[2] Amazon product data, Category: "Grocery and Gourmet Food", Balanced (see Figure 4)
[3] Annotated organic data set

Table 1: Data sets used to train our models

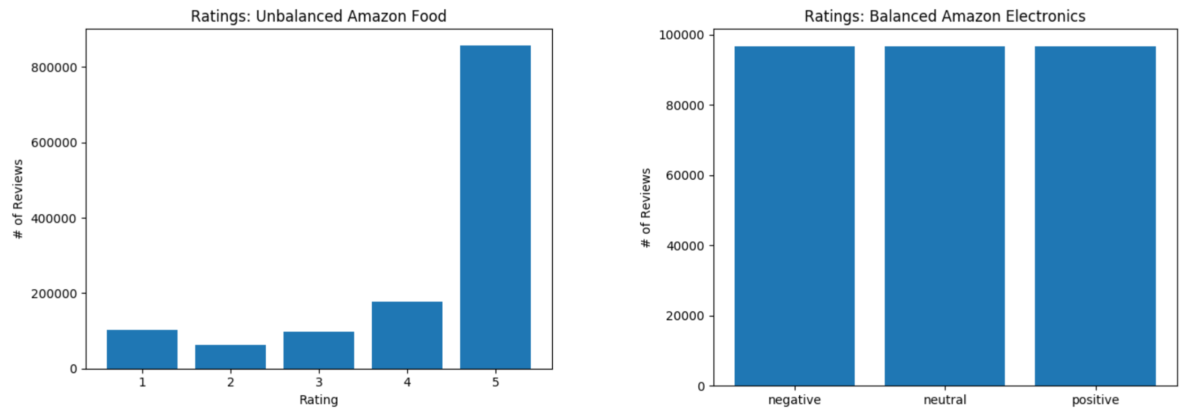
4.1 Training Data

The Amazon review data sets contain reviews from costumers and a rating from 1 to 5 stars. As our model only predicts three distinct sentiment classes we mapped 1 star to the negative sentiment, 3 stars to the neutral sentiment and 5 stars to the positive sentiment, in order to maximize the boundary towards the sentiment class 3. We balanced the data as seen in Figure 4 to have the same amount of negative, neutral and positive documents.

4.2 Word Embeddings

For the models using word embeddings, we created our own vocabulary by assigning an id to each word. Then we created a weight matrix containing for each word id i its embedding vector in its i-th

[2]https://gitlab.lrz.de/social-rom/overview/wikis/Organic-Dataset



(a) Before balancing (b) After balancing

Figure 4: Rating distribution of the Amazon food data set.

row. In our model, we then resolve the word ids to their corresponding word embedding vector by using the weight matrix as the weights of a non-trainable embedding layer. This mechanism helped us to dramatically reduce the size of the pre-processed model input since it eliminates the duplication of the word embeddings.

4.3 Elementary Discourse Units

We extracted the EDUs with the RST parser from Feng and Hirst (Feng and Hirst, 2014). This task is computationally intense and needs a lot of memory. We partitioned the data sets into 40 subsets and then ran the RST parser in 20 separate processes. This took ≈36 hours for the electronics data set.

4.4 Data padding

While the number of segments, as well as the number of words in a segment, varies for each document, our model requires its input matrices to have identical shape. Thus, we use zero-padding to create input matrices with identical shape. The width of the matrices corresponds to the number of words per segment and its height is determined by the number of segments per document. We chose both dimensions so that less than 5% of the reviews were truncated, while the rest was zero-padded.

5 Experiments

5.1 Training

We implemented our model with the Keras API of TensorFlow. Based on the results of previous work as well as by validating our own model, we set the hyperparameters in the following way: For the word level embedding, the heights of the convolutional kernels are set to $[3, 4, 5]$, with the hidden feature dimension being 100. In the attention layer, the dimension of the attention key h_a is set to 100. For all dropout layers, we used a dropout rate of 0.5. The model is trained with the Adam optimizer using the default settings for 8 epochs with batch size 512. Our w2v embeddings were pre-trained on the Google News corpus and for fastText we used the pre-trained English model.

5.2 Test Cases

Our main goal was to compare the performance of using EDU-wise segments against using sentence-wise segments. Therefore, we constructed the following test cases, from which we can successfully answer our primary question, while being able to evaluate different configurations of our model.

5.2.1 Segmentation Level

The first comparison is done between models with different segment levels. Those include sentence and EDU level segmentation, as mentioned previously. The models are trained on the Amazon food data set.

- **MIL-G-SEN** *Sentence-wise* segmentation. The input is embedded by fastText.
- **MIL-G-EDU** *EDU-wise* segmentation. The input is embedded by fastText.

With the same configuration trained on Amazon Electronics dataset, we obtained another set of models, named as **MIL-E-SEN**, and **MIL-E-EDU**.

5.2.2 Text Embedding

We compared the influence of different text embedding strategies. Therefore, we evaluated, additionally to the previously defined **MIL-G-SEN** (using fastText), models using word2vec and XLING on the Amazon food data set. Training with XLING takes a long time due to its usage of the TensorFlow Hub.

- **MIL-G-SEN-w2v**: Sentence-wise segmentation with *word2vec* features for each word.
- **MIL-G-SEN-xling**: Sentence-wise segmentation with *XLING* features for each segment.

To compare different segmentation levels and data sets we always used the fastText embedding.

5.2.3 Model Structure

To determine how the model‘s structure affects the performance of the model we created two baselines to compare against:

- **MIL-G-SEN-NA**: Multiple instance learning model of sentence-level with fastText embedding. This model does *not use attention* weights to combine the segment predictions. After generating the sentiment analysis probability distribution for each segment, the result is computed by averaging all segments.
- **NB-G**: *Naive Bayes* classifier based on bag of words. The model is regarded as the baseline model. When building the bag of words, the text is lemmatized and stop words are removed.

6 Results

The reported test scores with the suffix *"G"* are on the Amazon food data and the ones with the suffix *"E"* or on the Amazon food data. They are averaged over the three classes for all our results. In Table 2 a) we report the performance of using different embeddings. The word embeddings w2v and fastText perform very similar and outperform the sentence level embedding XLING. Similarly, we compare our default model MIL-G-SEN with the different model structures Naive Bayes and the model without attention in Table 2 b). This clearly shows the importance of the attention mechanism, to not average over each segment sentiment, but instead to learn based on the segment encoding which segments contain the most importance for the document sentiment. Visually this can be seen in the example of Figure 1. For our final presentation, we truncated 20% of the reviews to have less zero-padding when normalizing the size of our inputs. As seen in the here reported scores it turns out that this has a large negative effect on the model's performance: after only truncating less than 5% of the inputs, the accuracy of the models increased by 2% to 3% on average, and sentence segmentation outperforms EDU segmentation.

Method	F1-Macro	Accuracy
MIL-G-SEN (fastText)	0.795	**0.797**
MIL-G-SEN-w2v	**0.797**	0.79
MIL-G-SEN-xling	0.78	0.78

(a) Different embeddings

Method	F1-Macro	Accuracy
NB-G	0.693	0.695
MIL-G-SEN-NA	0.538	0.615
MIL-G-SEN	**0.795**	**0.797**

(b) Different model structures

Table 2: Comparison of different models on the Amazon food data set

For all our results the F1-Macro score of the neutral class always lacks behind the F1-Macro score for the positive and negative reviews as seen in the confusion matrix of Figure 5. This is common in other sentiment analysis models, as the extremes are simply easier to distinguish.

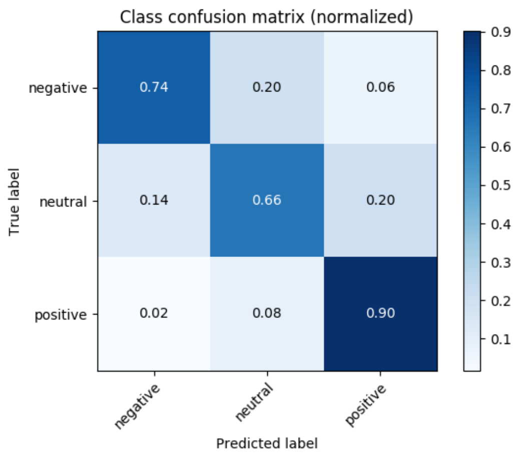


Figure 5: Confusion matrix of the F1-Macro scores for MIL-E-SEN

In Table 3 we compare the different segmentation levels, which was our main goal in the first place. Sentence-level segmentation outperforms EDU segmentation on both data sets. It is important to mention here that we ignored the labels 2 and 4 during our training, which makes those overall strong performances possible. In the Appendix (see Table 5) we tested our most important models again without removing those labels for comparison with other models on this data set.

Method	F1-Macro	Accuracy
MIL-E-EDU	0.748	0.75
MIL-E-SEN	**0.764**	**0.767**

(a) Amazon electronics

Method	F1-Macro	Accuracy
MIL-G-EDU	0.782	0.785
MIL-G-SEN	**0.795**	**0.797**

(b) Amazon food

Table 3: Comparison of the different segmentation levels

We annotated 300 sentences‘ EDUs of the organic data set to detect whether or not at least one triplet of attribute, entity and sentiment is distributed over more than one EDU in a sentence. As Figure 6 shows most triplets are distributed over more than one EDU (yes-bar), which supports our findings that EDUs are not the best candidate for document segmentation.

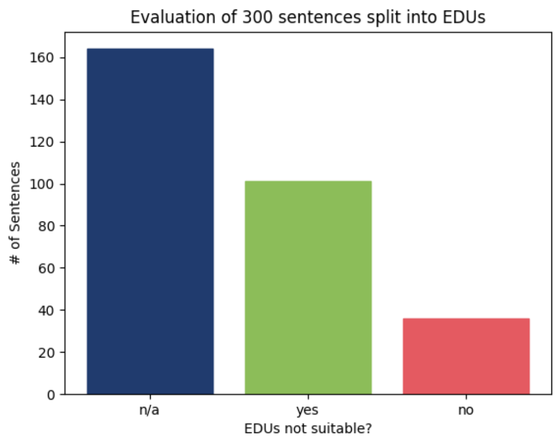


Figure 6: Evaluation whether or not EDUs are a suitable segment-level

7 Conclusion

As reported in the paper MILNET (Angelidis and Lapata, 2018) segmenting by EDUs performs worse than the sentence-wise segmentation. This finding is supported by our manual annotation on whether or not sentiment is distributed over more than one EDU. But even with the EDU wise segmentation, the multiple instance learning performed well, compared to our baseline. Overall the word embeddings performed better than the XLING embedding. As a next step, one could evaluate the segment-wise labels with the currently being annotated data set. For the EDUs one could also use the Rhetorical Syntax Tree and the relationships between the EDUs to perform smarter zero-padding of the segments and possibly even support the attention mechanism. One could also use the Rhetorical Syntax Tree to, e.g., increase the attention weight of the sub-sentence part containing a conjunction. This would follow the intuition that the most important sentiment always comes after the “but”.

References

Angelidis, Stefanos and Mirella Lapata. 2018. “Multiple instance learning networks for fine-grained sentiment analysis.” *Transactions of the Association for Computational Linguistics* 6:17–31.

Bojanowski, Piotr, Edouard Grave, Armand Joulin and Tomas Mikolov. 2017. “Enriching word vectors with subword information.” *Transactions of the Association for Computational Linguistics* 5:135–146.

Feng, Vanessa Wei and Graeme Hirst. 2014. “Two-pass discourse segmentation with pairing and global features.” *arXiv preprint arXiv:1407.8215* .

He, Ruining and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *proceedings of the 25th international conference on world wide web*. International World Wide Web Conferences Steering Committee pp. 507–517.

Horrigan, John. 2008. “Online Shopping, Pew Internet and American Life Project.” *Washington, DC Available at:< http://www. pewinternet. org/Reports/2008/Online-Shopping/01-Summary-of-Findings. aspx>[Accessed 8/8/2014]* .

Kim, Yoon. 2014. “Convolutional neural networks for sentence classification.” *arXiv preprint arXiv:1408.5882* .

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S Corrado and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. pp. 3111–3119.

Morey, Mathieu, Philippe Muller and Nicholas Asher. 2017. How much progress have we made on RST discourse parsing? A replication study of recent results on the RST-DT.

Rücklé, Andreas, Steffen Eger, Maxime Peyrard and Iryna Gurevych. 2018. “Concatenated power mean word embeddings as universal cross-lingual sentence representations.” *arXiv preprint arXiv:1803.01400* .

Socher, Richard, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*. pp. 1631–1642.

Yang, Zichao, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*. pp. 1480–1489.

Appendix

Detailed Performance of Models

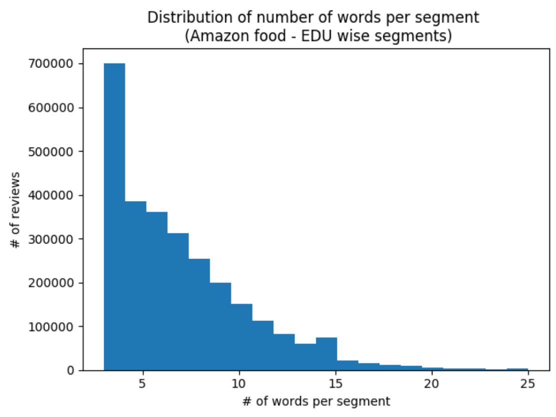
Method	Accuracy	F1-Macro	Negative-F1	Neutral-F1	Positive-F1
MIL-G-SEN	0.797	0.795	0.758	0.777	0.849
MIL-G-SEN-w2v	0.790	0.797	0.739	0.780	0.851
MIL-G-SEN-XLING	0.780	0.780	0.787	0.692	0.861
MIL-G-SEN-NA	0.615	0.538	0.276	0.676	0.662
MIL-G-EDU	0.785	0.782	0.790	0.814	0.895
MIL-E-EDU	0.750	0.748	0.759	0.667	0.818
MIL-E-SEN	0.767	0.764	0.780	0.682	0.830
NB-G	0.695	0.693	0.755	0.586	0.739

Table 4: Detailed test results

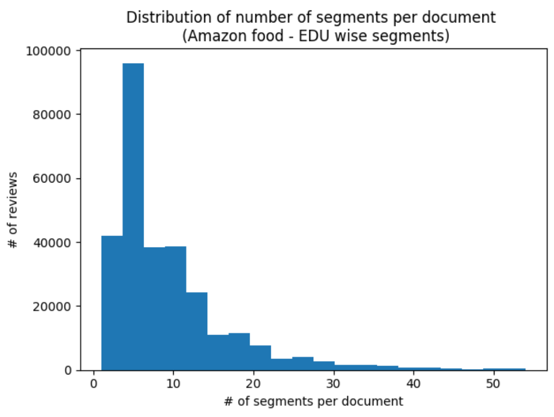
Method	Accuracy	F1-Macro	Negative-F1	Neutral-F1	Positive-F1
MIL-G-SEN	0.735	0.737	0.703	0.686	0.821
MIL-G-EDU	0.727	0.728	0.694	0.675	0.816
MIL-E-SEN	0.655	0.640	0.691	0.467	0.772
MIL-E-EDU	0.639	0.628	0.676	0.455	0.752

Table 5: Detailed test results with all 5 labels

Segment and Review Length Distribution

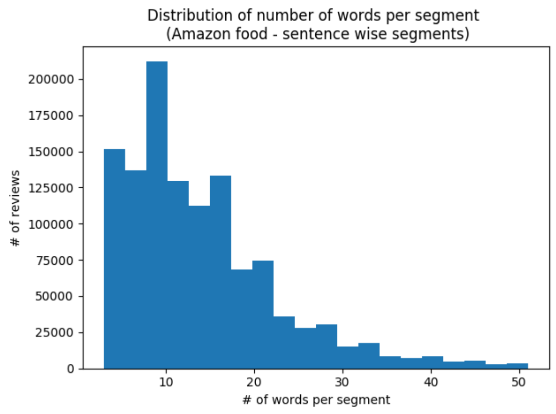


(a) Maximum segment length (in words)

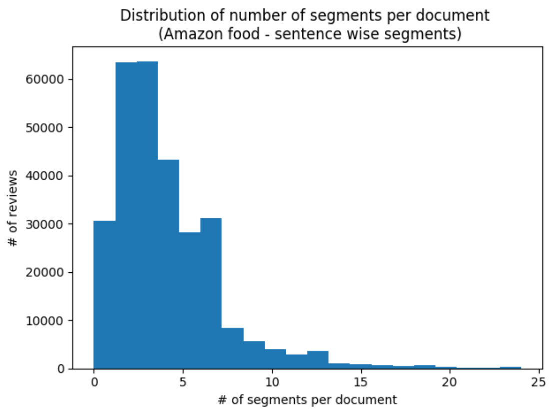


(b) Maximum review length (in segments)

Figure 7: Distribution of segment and review lengths for *sentence-wise* segmentation of Amazon Fine Foods



(a) Maximum segment length (in words)



(b) Maximum review length (in segments)

Figure 8: Distribution of segment and review lengths for *EDU-wise* segmentation of Amazon Fine Foods