Novel Views on Novels: Embedding Multiple Facets of Long Texts

Lasse Kohlmeyer
lasse.kohlmeyer@student.hpi.de
Hasso Plattner Institute
University of Potsdam, Germany

Tim Repke
tim.repke@hpi.de
Hasso Plattner Institute
University of Potsdam, Germany

Ralf Krestel
ralf.krestel@acm.org
ZBW Leibnitz Information Centre for Economics
Kiel University, Germany

ABSTRACT

Novels are one of the longest document types and thus one of the most complex types of texts. Many NLP tasks utilize document embeddings as machine-understandable semantic representations of documents. However, such document embeddings are optimized for short texts, such as sentences or paragraphs. When faced with longer texts, these models either truncate the long text or split it sequentially into smaller chunks. We show that when applied to a fictional novel, these traditional document embeddings fail to capture all its facets. Complex information, such as time, place, atmosphere, style, and plot is typically not represented adequately.

To this end, we propose *lib2vec* which computes and combines multiple embedding vectors based on various facets. Instead of splitting the text sequentially, *lib2vec* splits the text semantically based on domain-specific facets. We evaluate the semantic expressiveness using human-assessed book comparisons as well as content-based information retrieval tasks. The results show that our approach outperforms state-of-the-art document embeddings for long texts.

ACM Reference Format:
Lasse Kohlmeyer, Tim Repke, and Ralf Krestel. 2021. Novel Views on Novels: Embedding Multiple Facets of Long Texts. In *WI-IAT '21: Web Intelligence and Intelligent Agent Technology December 14–17, 2021, Melbourne, Australia.* ACM, New York, NY, USA, 8 pages. https://doi.org/00.0000/0000000.0000000

1 INTRODUCTION

Reading is one of the main strategies to obtain new information. However, the availability and the amount of information are continuously growing. This is not only the case for user-generated content, scientific papers, or news, but also for books [13, 41]. For example, in 2019, 78 746 new books were published in Germany [28] and 1 600 000 books were self-published in the USA in 2018 [8]. These numbers clearly illustrate the so-called *information overload problem* [3, 48] which applies to book publishers and readers alike. Automatic methods can assist human decision making, for example, book publishers who have to select books from a large number of submitted manuscripts by judging whether they align with their program and potential commercial success [6].

WI-IAT '21, December 14–17, 2021, Melbourne, Australia
© 2021 Association for Computing Machinery.
ACM ISBN 000-0-0000-0000-0…$15.00
https://doi.org/00.0000/0000000.0000000

Machine-understandable representations of books are the basis for such automated approaches. Numerical representations can be used to calculate similarities between books as well as for clustering, classification, or content-based recommender systems. Modern document embeddings are usually designed to encode shorter texts. For example BERT [11] is limited to 512 tokens, Longformer [7], which is designed for long sequences, is limited to 4 096 tokens, and doc2vec has a limit of 10 000 while books may easily exceed 100 000 tokens. Furthermore, there are hardly any publications that process the full-text of books holistically [1]. However, full-text is one of the key characteristics of a book and should be considered when encoding a book.

To this end, we propose *lib2vec*[1] which embeds subsets of words of a book individually using state-of-the-art embedding algorithms. These subsets, which we associate with *facets*, represent different angles, views, or aspects of a book. In the context of novels, which is the focus of this study, possible facets of interest are the plot, where and when the plot is set, the novel's atmosphere and style. To compute these facet embeddings, we identify words in the text corresponding to facets and construct pseudo-documents. Each facet embedding can be used individually for applications that need to compare aspects of a novel. We also examine strategies to combine facet embeddings into a single embedding vector for a novel.

We evaluate the performance of our approach on different tasks with multiple English and German corpora of novels and compare it to a number of state-of-the-art embedding models. Our experiments show that *lib2vec* is able to significantly outperform other models in determining author and genre similarity as well as identifying books belonging to the same series, such as Harry Potter sequels. In addition, we evaluate our embedding on genre and rating prediction of novels. Furthermore, we introduce the *BoCo* dataset with similarities between 20 books under different aspects annotated by 81 domain experts.

Our contributions are: (1) We present *lib2vec*, a framework which computes and combines multiple embedding vectors based on various facets. Instead of splitting the text sequentially, *lib2vec* splits the text semantically based on domain-specific facets. (2) *lib2vec* provides memory and time-efficient computation of embedding vectors for long texts. (3) *lib2vec* allows the comparison of books with respect to different facets. (4) We compare *lib2vec* to competitive baselines in a wide range of experiments. (5) We introduce a new benchmark dataset to evaluate book similarities.

[1]Based on the Latin word for book: liber.

2 RELATED WORK

Semantically meaningful book representations are closely related to word and document embeddings. Word embeddings, such as word2vec [32], quickly gained popularity, as they provide a dense vector representation for words that can be used to, e.g., calculate the semantic similarity between words. Grayson et al. [16] apply word2vec on 19th-century novels to investigate to which extend quantitative literary analysis can profit from the semantic similarities between word embeddings in their approach *novel2vec*. A big difference to our *lib2vec* framework is that no novel representation is obtained by *novel2vec*. However, such an encoding of a book is calculated by Anvari and Amirkhani [4]. Their approach *book2vec* uses the reading histories of users to model sequences of books which serve as input for the word2vec algorithm. In contrast to *book2vec*, *lib2vec* is applied on the texts of novels, which is not dependent on user-data such as reading histories.

To utilize word embeddings in full-text applications, [25] introduced paragraph embeddings for short texts. However, their input sequence is limited and longer texts need to be truncated, which leads to a loss of information and poor performance for long, heterogeneous texts. Modern, transformer-based document embedding algorithms have a much higher limit by utilizing an internal self-attention mechanism to capture contextual information of entire documents [9, 22, 36]. However, these models are typically task-specific and the attention mechanism has very high computation and memory requirements. For example, the most recent Longformer model requires multiple powerful GPUs with 48GB RAM to cover documents of up to 32K tokens [7].

P-SIF was introduced to embed longer documents [17]. It considers the topical structure of documents by concatenating word embeddings over the topic distribution of words. P-SIF has similarities to our work, since in both algorithms embeddings for a fixed number of selected groups of words are computed. However, in contrast to our work, only topical information of a text is utilized by P-SIF and not other kinds of information such as style or location. Furthermore, P-SIF is evaluated on news articles or reviews that are much shorter than novels.

Other downsides of existing document embedding approaches are the poor performance in classification tasks and the lack of interpretability of the latent vector space. Both are addressed by Unnam and Reddy [45], who utilize words with high discriminatory power as dimension features for the latent space of documents which improves classification performance and interpretability. We show that by embedding multiple facets our approach addresses these issues as well.

Multi-view learning or multi-faceted embeddings refer to the same family of approaches and are often motivated by the polysemy of words [34]. Multi-faceted text embeddings often use different data sources. These different data sources are, e.g., utilized to achieve a higher quality for general word embeddings [30] or domain-specific word embeddings [39]. Thereby, embeddings of the same word based on different data sources serve as independent facets. The facets are combined by dimensionality reduction based on neural networks [30] or PCA [39]. Nevertheless, a minority of works create facets from the same data source. In this area, Neelakantan et al. [34] train embeddings that capture the polysemy of words. Starting with the word to be embedded, the authors cluster the surrounding context words into word sense clusters. Afterward, they embed each of the sense clusters separately to form the different facets. Very similarly, *lib2vec* uses the same text data to form different facets. However, instead of word embeddings, *lib2vec* calculates document embeddings, and instead of word context clustering to obtain sets of words, it selects words by leveraging annotations such as part-of-speech tags.

To create different facets of documents, other information such as click or interaction data can be used in addition to text data [27, 42]. Thus, most text-related multi-faceted embeddings leverage facets based on auxiliary data apart from Risch et al. [40]. To encode book synopses, they embed three different facets: time, place, and plot. While for the time embedding they trained a classifier that assigns a year to each synopsis, for the location they average the word embeddings of all location words. Only the representation of the plot is obtained by doc2vec. In contrast, *lib2vec* computes all facet embeddings in a unified way, using one and the same document embedding algorithm, e.g., doc2vec. Instead of having to train an additional classifier or use other external information, the information contained in the text is leveraged. Thus, new facets can be constructed with less effort. Another difference to Risch et al. [40] is the selected data, as synopses are much shorter than novels and provide other possibilities for facets.

Besides multi-faceted embeddings, there are other ways to encode novels. Many of these approaches are based on hand-crafted features such as readability [35], stylometric features for authorship attribution [2], as well as token/type ratios, word frequencies, or the number of female characters within a text [1]. A disadvantage of hand-crafted features is that they are often very domain-specific and are difficult to transfer to other scenarios. In contrast to hand-crafted features, *lib2vec* generates different facets in an unsupervised manner, based on annotations. While they are also not domain-independent, they can be customized with less effort, as only relevant annotations need to be identified.

Maharjan et al. [31] show that hand-crafted features such as character n-grams are often outperformed by neural-network-based approaches. The authors obtain full-text representations of books based on various strategies, e.g., feature-based or based on document embeddings. They created a Goodreads [2] dataset which contains annotations for success, in form of ratings, and genre. We use this dataset for testing rating and genre classification. To tackle the input sequence limit of document embedding algorithms, the authors represented each sentence as the average of its word embeddings and then fed chunks of 128 of these vectors into a multi-layer RNN to get a representation for the whole text as a result. In contrast, *lib2vec* does not apply chunking, since it leads to a loss of information and therefore to poor quality of embeddings. The experiment by Maharjan et al. [31] is adopted by Khalifa and Islam [21], who created book encodings by sentence embeddings incorporating readability scores. Again, our approach differs, because our strategy for dealing with long documents is less affected by the loss of information when different sentence embeddings are aggregated.

[2]https://www.goodreads.com/

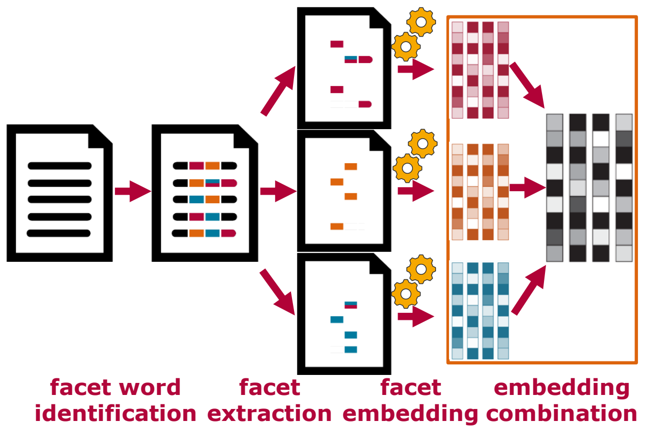

Figure 1: Overview of processing steps of *lib2vec*.

3 MULTI-FACETED DOCUMENT EMBEDDINGS

Embeddings for long documents are typically realized by splitting them into smaller parts following either a truncation-based, chunk-based, or a facet-based strategy. Truncation-based approaches are using only the first n words of a text, where n corresponds to the input sequence limit of the embedding algorithm. Chunk-based approaches split a text into several segments of size n with $n \leq$ input sequence limit. Then, each segment is embedded separately and then combined, for example via summation. Both these strategies loose crucial information either by ignoring large parts of the text (truncation) or by averaging intermediate results (chunking). To overcome these disadvantages, we propose *lib2vec* which follows the idea by Risch et al. [40] and divides a text into several facets, instead of chunks. The novel core idea of *lib2vec* is to split long texts not sequentially based on document position but rather selectively based on semantic facets

We define facets as subsets of words following semantic categories. In this work, these categories are oriented toward the perspectives of literary analysis such as atmosphere or style. Thereby, each category includes certain words, for example, the style facet includes stopwords since their usage is a good indicator for different styles. Besides, words that appear in one facet may also appear in another facet. Compared with truncation-based approaches, facet-based approaches make use of more words of a text. While the number of chunks differs for each text, the number of facets is constant for any text. In addition, facets are semantically meaningful, while chunks only provide information about the position within a text.

3.1 Processing Pipeline of lib2vec

Figure 1 illustrates the overall concept for our multi-faceted embedding approach: identifying candidate words, constructing pseudo-documents, and computing embeddings.

Facet Word Identification. In the first step, documents are preprocessed with sentence splitting, tokenization, lemmatization, POS-tagging, named entity recognition (NER), and semantic word dictionary lookups. Each book in a corpus is annotated automatically, using SpaCy [20], Heideltime [43], the German semantic dictionary GermaNet [18, 19], and the English semantic dictionary WordNet [33].

Facet Extraction. In this step, words in a document are assigned to none, one, or multiple of the facets *location*, *time*, *style*, *atmosphere*, or *plot*. Additional facets can be defined based on the domain if necessary. For each facet, a pseudo-document is generated containing all assigned words.

Facet Embedding. Each pseudo-document consists of only a fraction of the words of the original book. This allows to compute facet-specific embeddings using existing document embedding approaches. We opted for using doc2vec since it has the largest input sequence limit. In case the pseudo-document exceeds this limit, the facet is truncated.

3.2 Facet Definitions

As described earlier, we use the tagged documents to construct pseudo-documents for each facet. In this section, we describe in more detail how these pseudo-documents are compiled.

Location. The location facet is based on tokens that are tagged as geopolitical locations and buildings by SpaCy NER. Multiple appearances of a word are kept to preserve their frequency, as intermittently reappearing places may be more important to the book. A downside of NER is, that unnamed places such as *ocean*, *garden*, or *street* are not included. Thus, we use GermaNet or WordNet to identify location descriptors.

Time. Similar to location, we use NER to construct the pseudo-document for the time facet. Although this covers months, seasons, or year numbers, phrases referring to historical events such as revolutions or wars are not tagged by standard NER tools. Thus, we again utilize GermaNet or WordNet to extend the scope to include events and abstract time descriptors.

Style. Prior studies have shown correlations between authorship attribution, the style of a text, and the usage of stopwords [2, 5, 44]. Following these findings, we construct the style facet based on all stopwords found by a POS-tagger. Since stopwords appear very frequently, we expect this pseudo-document to grow quickly. Although this might be problematic considering sequence length limitations of embeddings, we assume that truncated stopwords are already present in the limited sequence and the potential loss of information is small.

Atmosphere. Adjectives and adverbs are common linguistic indicators of the atmosphere of a text since they describe the properties of objects, people, or emotions. In addition, affective or sensual words such as *gloomy*, *warm*, *smell*, or *howl* also have a great influence on the atmosphere [44]. In order to take these affective and sensual words into account, we consider nouns, verbs, and adjectives corresponding to emotions and feelings as identified by GermaNet or WordNet.

Plot. We utilize the basic assumption that the plot of a novel describes the actions between characters [23, 26]. Such actions are expressed by verbs and adverbs. Thus, we construct the pseudo document for the plot facet based on words with these POS-tags.

Raw. In addition, we also include a full-text facet containing the beginning of a book without any filtering apart from the truncation after the maximum sequence length. In contrast to other facets, the raw facet preserves continuous text and word order which may improve the results if the applied document embedding algorithm can take advantage of such textual continuities.

The assignement of words to individual facets is done based on POS- and NER-annotations. For the time and location facets, the disambiguation is done by the named entity recognition model. For the other facets, the POS-annotations are used. Single words may appear in multiple facets, for example, "sunset" may be used as a time ("before sunset") or as a location ("Sunset Blvd") descriptor in the same book. We rely on state-of-the-art POS and NER tools to correctly assign POS- and NER-tags. These tools are not perfect. Improving them should also boost the performance of *lib2vec* further.

3.3 Combination of Facets

In the previous sections, we described the construction of pseudo-documents for facets. For practical reasons and in order to compare our approach to other document embedding methods, we need to combine the facet embeddings into a single vector. Rettig et al. [39] cover a number of approaches of aggregating embedding vectors for their multi-faceted word embeddings. The naïve approach to combine facet embeddings is to use the arithmetic mean (AVG) or an element-wise sum of the individual vectors. Gittens et al. [15] have empirically shown that the summation of word and document embeddings can retain aspects of two underlying embedding vectors. Another approach is to simply concatenate facet embedding vectors (CON). This has the advantage that facet embeddings can have a different dimensionality and the information captured by each facet is still available in the resulting vector. We also experimented with other dimensionality reduction methods, such as autoencoders and concatenation followed by principal component analysis, but results were worse compared to simple AVG and CON.

4 EXPERIMENTAL SETUP

In this section, we provide an overview of which datasets we use for our evaluation, how we mitigate the influence of potential biases in our experiments, and describe the baselines we compare our approach to.

4.1 Datasets

Key characteristics of the datasets used are provided in Table 1. The median is reported for the average number of (unique) words per book and the mean for the average number of books per author. An overview of the average coverage for each facet is provided in Table 2. The average coverage is calculated by averaging the ratio of facet tokens and the number of total tokens per document over the corresponding corpus.

Corpus of German-Language Fiction (CGF). The Corpus of German Fiction by Fischer and Strötgen [14] consists of 3 219 German prose books published between 1840 and 1930. For our series classification task, we identified 208 books that were part of a series or sequels.

Table 1: Overview of Dataset Characteristics

Dataset	CGF	DTA	MGG	BoCo
Language	GER	GER	EN	EN
# Books	3.2K	459	1K	20
# Words	204M	26M	17M	4M
# Unique Words	26M	4M	3M	198K
$\frac{\text{\# Words}}{\text{per Book}}$	46K	44K	17K	145K
$\frac{\text{\# Unqiue Words}}{\text{per Book}}$	7.2K	7.5K	2.8K	8.9K
$\frac{\text{\# Books}}{\text{per Author}}$	5.7	2.1	2.1	1.2
$\frac{\text{\# Books}}{\text{per Series}}$	2.5	3.2	–	–

Table 2: Overview of Average Facet Coverage Values

Facet	CGF	DTA	MGG	BoCo
Location	0.012	0.015	0.006	0.004
Time	0.007	0.006	0.010	0.009
Style	0.479	0.465	0.470	0.505
Atmosphere	0.144	0.142	0.106	0.110
Plot	0.183	0.179	0.167	0.184
Raw	1.000	1.000	1.000	1.000

Deutsches Text Archiv (DTA). The DTA corpus contains 765 German works of belles-lettres published between the 16th and 20th century [10]. We exclude texts published before 1800, as the German language has changed significantly since then, leaving 459 texts. In this corpus, we were able to identify 176 books which are part of a series.

Gutenberg and Genres (MGG). Maharjan et al. [31] created the MGG corpus consisting of 1 003 full-texts of English books to evaluate success and genre prediction methods. It contains genre information and binary labels of success based on Goodreads ratings. A book is labeled successful if it has an average rating of more than 3.5 stars from at least ten people.

Book Comparison (BoCo). This dataset consists of 20 popular English books selected from Project Gutenberg.[3] We conducted a survey with 81 participants who were asked to asses which two books in a triplet of books are more similar with regard to when and where they are set, as well as their atmosphere and plot. In total, we were able to gather ratings on 527 triplets with a moderate annotator agreement of a Fleiss' Kappa Score of 0.55.

4.2 Removal of Possible Biases

In preliminary experiments, we saw a significant correlation between the length of a text and the similarity score for doc2vec and average word2vec representations. Our *lib2vec* approach was far less affected by the length of books. Nevertheless, to allow for a fair comparison between approaches, we run all experiments with books of comparable length. To exclude length as a distinguishing feature, books are divided into three categories: short, medium, and

[3]www.gutenberg.org

long, with threshold values determined by quantiles of 0.33 and 0.66 of the token counts.

For experiments on identifying series of books or books by the same author, we remove words that may provide a clear distinction. For example, character names of a sequel may not appear in other books and thus render identifying books of the same series a trivial task. We do not explicitly identify character names in the text, but they are removed implicitly by filtering the vocabulary based on document frequency. We remove any words with a document frequency below a certain threshold defined for each dataset. Although this significantly reduces the size of the overall vocabulary, document length distributions are barely affected.

4.3 Baselines

We compare *lib2vec* to various traditional and state-of-the-art baselines. As the most simple baseline, we choose a bag-of-words (BoW) representation limited to a vocabulary size of 30 000 words. Furthermore, we use standard word2vec embeddings trained on each corpus. We calculate the arithmetic mean of all words embeddings in a book to get a single vector representation for the entire book. In addition, we compute representations of books using doc2vec [25]. Preliminary experiments with pre-training the doc2vec model yielded worse results, therefore we train the doc2vec model on each corpus separately. We also use more recently published models, namely BERT [11], RoBERTa [29], XLM [24], fine-tuned for sentence representation by SentenceTransformers [38]. These approaches are limited by the maximum sequence length, however Khalifa and Islam [21] have shown that the first 1,000 sentences can be sufficient to encode books. We applied the sbert sentence transformer to the first 1000 sentences of each book and averaged these vectors to encode each book. Further, we compare to the partitioned word averaging model P-SIF [17] as a state-of-the-art document embedding algorithm which is designed particularly for long documents.

4.4 Hyper-Parameters

The *lib2vec* embeddings are trained using the Gensim [37] implementation of doc2vec. The following hyperparameters are chosen for the trained embeddings by manual tuning: An embedding dimension of 300 is considered because this is a common choice for word and document embeddings [39]. Besides, a preliminary experiment was performed with different dimension sizes for doc2vec on the CGF dataset. The result confirms, that a dimension size of 300 is the smallest but best performing choice for book similarity tasks. A similar preliminary experiment was performed to obtain a good choice for the window size which resulted in a size of 5. This value is also suggested by Grayson et al. [16] for semantic relationships. Each model is trained for 20 epochs. All non-deterministic implementation options such as the usage of multiple workers are disabled. For the other options, the default values are preserved. No pre-trained models or further hyperparameter optimization are involved to make results comparable. The same hyper-parameter setup is used for all baselines. All experiments are executed on a Windows midrange machine with 16 GB RAM and 48 GB swap partition, an AMD Ryzen 5 3600 CPU with 12 cores, and no GPU. Table 3 shows the runtime in minutes each model needs for training/fine-tuning on the various corpora. Please note, that the large models, such as BERT, already come pre-trained. The time for this pre-training is not considered here. Nevertheless, *lib2vec* is considerably faster compared to these large models.

Table 3: Runtime in Minutes

	CGF	DTA	MGG	BoCo
BoW	95	21	2	1
AVG W2V	1026	49	23	7
doc2vec	248	47	22	8
BERT	880	205	28	23
RoBERTa	1014	230	32	56
XLM	1006	192	30	104
P-SIF	95	17	6	2
lib2vec	438	13	20	1

5 EVALUATION

We evaluate our approach on various tasks: unsupervised (Subsections 5.1,5.3 and supervised (Subsection 5.4). For all experiments, we only use the documents of each corpus to train the doc2vec embedding for each facet for our approach. In contrast to models such as BERT, which need to be trained on millions of tokens, *lib2vec* achieves very good results with only a fraction of training data.

5.1 Book Similarity

As a first experiment, we use an information retrieval setting to measure how well embeddings encode similarities of books from the same author, same genre, and same book series. For each book, we look at the 10 nearest neighbors based on cosine similarity in the respective embedding space and compute precision@10 for the three tasks (see Table 4; scores closest to upper bound are highlighted in bold). Note that not all datasets have series or genre annotations and are therefore excluded from some of the experiments. Given that we do not have 10 positive examples for all tasks

Table 4: Book Similarity: Precision@10 for Author, Series, and Genre Prediction Task

	Author			Series		Genre
	CGF	DTA	MGG	CGF	DTA	MGG
BoW	0.28	0.18	0.13	0.18	0.31	0.44
AVG W2V	0.26	0.17	0.12	0.15	0.31	0.50
doc2vec	0.43	0.21	0.22	0.18	0.31	0.50
BERT	0.19	0.04	0.05	0.13	0.09	0.34
RoBERTa	0.06	0.04	0.04	0.06	0.06	0.29
XLM	0.11	0.04	0.05	0.09	0.07	0.34
P-SIF	0.31	0.17	0.14	0.17	0.30	0.52
lib2vec AVG	0.53	0.22	**0.23**	**0.19**	0.31	0.55
lib2vec CON	**0.62**	**0.23**	0.20	**0.19**	**0.32**	**0.79**
Upper bound	0.74	0.31	0.34	0.21	0.32	1.00

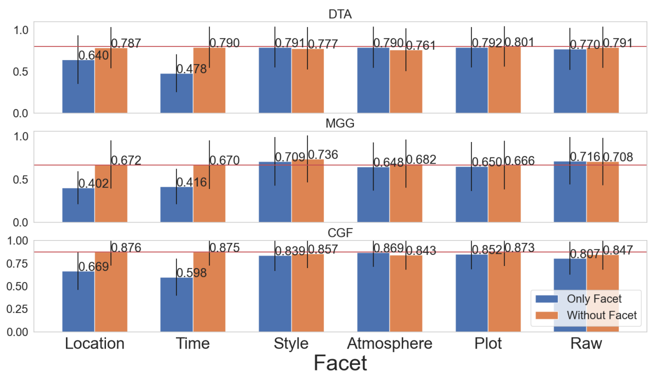

Figure 2: NDCG results for identifying authors in the book similarity task using only single facets or removing single facets.

and classes, we also report the upper bound of maximal achievable precision@10. Our *lib2vec* embedding consistently performs best and significantly outperforms state-of-the-art embeddings as well as traditional vector representations. For the genre classification task, we use the MGG corpus (the others don't contain genre annotations).

5.2 Ablation Study

In order to quantify the importance of each facet in our book similarity experiment, we carried out an ablation study on all three datasets. We use the same experimental setup as described in the previous section and measure NDCG. Results for the book similarity task for authors are shown in Figure 2. Here, instead of using all facets, we either use only one (blue) or all but one (orange) facet. Horizontal red lines indicate the respective score when using all facets, vertical lines are error bars based on repeated runs. Our results show, that the location and time facets have the least influence on author similarities. As expected, the style facet on its own is a strong identifier for authors, while removing the atmosphere facet has the most negative effect. The plot facet on its own also performs well, however removing it does not hurt the the results since other facets seem to capture this information as well. There are no statistically relevant conclusions on the influence of the raw facet. Although some scores with only one facet (or one facet left out) are higher than using all facets, using all available information generally leads to more robust results.

5.3 Book Comparison

In the previous experiment, we evaluated the different embeddings with respect to metadata associated with each book. To evaluate the facets more directly, we created the BoCo dataset containing manual book similarity annotations with respect to different aspects. Each expert rated which two books out of a set of three books are more similar to each other with respect to when and where the story is set (time, location), as well as their plot and atmosphere similarity. Each triplet has multiple annotations and we use the majority vote in cases where the annotators disagreed.

Table 5: Book Comparison: Precision of Baselines and *lib2vec* on the Human-Annotated BoCo Dataset

Algorithm	Location	Time	Plot	Atmosphere	AVG
BoW	0.49	0.46	0.46	0.48	0.47
AVG W2V	0.46	0.59	0.46	0.42	0.47
BERT	**0.55**	0.57	0.57	**0.58**	0.57
RoBERTa	0.45	0.39	0.49	0.50	0.47
XLM	0.43	0.36	0.47	0.41	0.42
P-SIF	0.41	0.42	0.38	0.37	0.39
lib2vec AVG	0.54	**0.64**	**0.59**	**0.58**	**0.58**
lib2vec CON	0.46	0.49	0.38	0.40	0.44

Table 6: Book Classification: Weighted F1-Scores for Classification of High-Rated Books and Genres

Algorithm	Rating	Genre
Maharjan et al. [31]	*0.72*	–
Khalifa and Islam [21]	*0.72*	–
BoW	0.61	0.19
AVG W2V	0.70	0.59
doc2vec	0.71	0.56
BERT	0.70	0.49
RoBERTa	0.69	0.41
XLM	0.71	0.47
P-SIF	0.70	0.55
lib2vec AVG	0.72	0.63
lib2vec CON	**0.76**	**0.89**

Table 5 lists the precision of different models regarding the agreement with human annotators. We consider only precision because the annotations do not provide information about false negatives which are required for recall. In order to rate the similarity, we use the respective document embeddings and calculate the cosine similarity between the three books in a given triplet. We then count for how many triplets a model agrees with the human annotations. For *lib2vec*, we use facet embeddings where applicable. Surprisingly, traditional embeddings and even bag-of-words are able to beat state-of-the-art transformer model P-SIF, which was especially designed for long texts. Overall, BERT and *lib2vec* perform best and yield very similar results across all facets. Also note that *lib2vec* AVG outperforms *lib2vec* CON only on th BoCo dataset. The reason could be that this is by far the smallest dataset with only 20 books.

5.4 Book Classification

We also evaluate the performance of different embeddings in a document classification setting as a downstream application. The MGG dataset [31] contains user ratings and genre information. Only 19 out of $\approx$ 1000 books books have two genres assigned, all other books only one, presumable the main genre. Books with multiple labels are included as duplicates in the corpus and result in the same embedding. We use the same binary classification setup as [31] and [21], which uses the weighted F1-score to measure performance. For a fair comparison, only the input representations

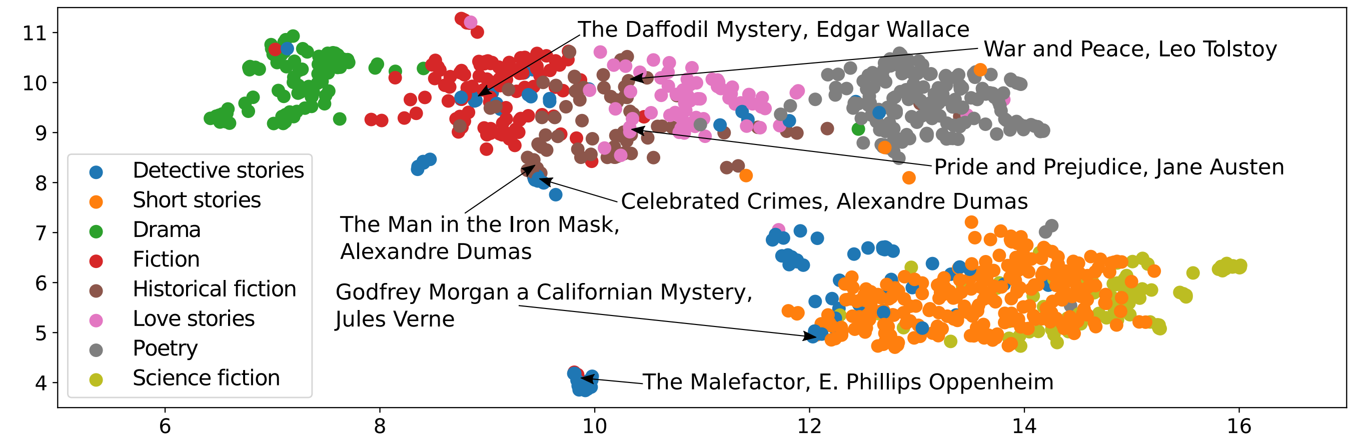

Figure 3: UMAP projection of the facet *plot* of *lib2vec* trained on the MGG corpus.

are changed; the classifier and the training dataset is identical. We train an SVM with RBF-kernel on the LitRec dataset [47] using the different representations. The dataset contains 3 458 full-texts of English books written by 1 109 authors from Project Gutenberg which were rated by 35 507 users. Table 6 shows the classification performance for rating and genre prediction (The largest values are highlighted in bold; italic values were taken from the corresponding publications). The classifier using our *lib2vec* embedding is able to outperform all classifiers using the baseline representations as well as the representations from related work, which are especially tailored toward this task. More sophisticated classifiers on top of the different representations would probably perform better. We deliberately chose a rather simple SVM to focus on the effect that the different representations have on the classification results.

5.5 Qualitative Evaluation

We also performed an extensive qualitative evaluation of the *lib2vec* model with respect to aggregated representation and individual facet embeddings in collaboration with domain experts from literary sciences. Figure 3 depicts a UMAP visualization of the embedding space of the "plot"-facet for the MGG corpus. The different colors indicate the genre of the novels. Although the genre information was not given to the model as an input, the representations learned by *lib2vec* for the "plot"-facet reflect different genres remarkably well. Book plots from the same genre are embedded very close to each other as can be seen by the colors grouping nicely in Figure 3. Note that the plots of the detective stories are scattered across multiple genre clusters and that, e.g., historical fiction has some overlap with love stories.

6 CONCLUSIONS & FUTURE WORK

Traditional embedding algorithms have an input sequence limit for the efficient encoding of documents. Long texts, such as novels, often exceed these limits. Therefore, not all available information is used when embedding long documents, or the long documents are split sequentially into smaller chunks. Both solutions result in sub-optimal performance on a variety of tasks regarding long documents. We introduced *lib2vec*, a multi-faceted document embedding framework that splits long documents not sequentially but semantically based on domain-specific facets. It provides semantically more meaningful document representations than truncation- or chunk-based methods. Our approach outperforms baselines, such as doc2vec, BERT, or P-SIF for corpora that contain large- and medium-sized novels on classification tasks, rating prediction, and assessing similarities.

The most interesting question for future work is how to incorporate additional knowledge into our model, e.g., by pre-training and/or fine-tuning. Large language models encode semantic knowledge that we aim to utilize for facet computation in the future. We further want to test *lib2vec* in other domains with other facets, e.g., for long scientific texts, or legal documents. Also, in the literary domain, there are facets we didn't explore yet, such as readability [12] or literariness [46]. Furhter, training an end-to-end framework sounds intriguing but it is unclear how to do this in an unsupervised way, since there is no annotated data yet containing facet information.

REFERENCES

[1] Haifa Alharthi and Diana Inkpen. 2019. Study of Linguistic Features Incorporated in a Literary Book Recommender System. In *Proceedings of the ACM/SIGAPP Symposium on Applied Computing*. ACM, New York City, NY, USA, 1027–1034. https://doi.org/10.1145/3297280.3297382
[2] Haifa Alharthi, Diana Inkpen, and Stan Szpakowicz. 2018. Authorship Identification for Literary Book Recommendations. In *Proceedings of the International Conference on Computational Linguistics (COLING)*. ACL, Stroudsburg, PA, USA, 390–400.
[3] Haifa Alharthi, Diana Inkpen, and Stan Szpakowicz. 2018. A Survey of Book Recommender Systems. *Journal of Intelligent Information Systems* 51, 1 (2018), 139–160. https://doi.org/10.1007/s10844-017-0489-9
[4] Soraya Anvari and Hossein Amirkhani. 2018. Book2Vec: Representing Books in Vector Space Without Using the Contents. In *Proceedings of the International Conference on Computer and Knowledge Engineering (ICCKE)*. IEEE, New York City, USA, 176–182. https://doi.org/10.1109/ICCKE.2018.8566329
[5] R. Arun, V. Suresh, and C.E. Veni Madhavan. 2009. Stopword Graphs and Authorship Attribution in Text Corpora. In *Proceedings of the IEEE International Conference on Semantic Computing (ICSC)*. IEEE, New York City, USA, 192–196. https://doi.org/10.1109/ICSC.2009.101
[6] Vikas Ganjigunte Ashok, Song Feng, and Yejin Choi. 2013. Success with style: Using writing style to predict the success of novels. In *Proceedings of the Conference*

on Empirical Methods in Natural Language Processing (EMNLP). ACL, Stroudsburg, PA, USA, 1753–1764.

[7] Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The Long-Document Transformer. *arXiv preprint* 2004.05150 (April 2020), 1–17.

[8] Bowker. 2019. Self-Publishing in the United States, 2013-2018. Print and Ebooks. https://media2.proquest.com/documents/bowker-selfpublishing-report2019.pdf

[9] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating Long Sequences with Sparse Transformers. *arXiv preprint* 1904.10509 (2019), 1–10.

[10] Deutsches Textarchiv. 2021. Grundlage für ein Referenzkorpus der neuhochdeutschen Sprache. https://www.deutschestextarchiv.de/

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*. ACL, Stroudsburg, PA, USA, 4171–4186. https://doi.org/10.18653/v1/n19-1423

[12] Lijun Feng, Martin Jansche, Matt Huenerfauth, and Noémie Elhadad. 2010. A comparison of features for automatic readability assessment. In *Proceedings of the International Conference on Computational Linguistics (COLING)*. ACL, Stroudsburg, PA, USA, 276–284.

[13] Jonathan Fink-Jensen. 2015. Book Titles per Capita. https://datasets.iisg.amsterdam/dataset.xhtml?persistentId=hdl:10622/AOQMAZ

[14] Frank Fischer and Jannik Strötgen. 2017. Corpus of German-Language Fiction. https://doi.org/10.6084/m9.figshare.4524680.v1

[15] Alex Gittens, Dimitris Achlioptas, and Michael W. Mahoney. 2017. Skip-Gram - Zipf + Uniform = Vector Additivity. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*. ACL, Stroudsburg, PA, USA, 69–76. https://doi.org/10.18653/v1/P17-1007

[16] Siobhán Grayson, Maria Mulvany, Karen Wade, Gerardine Meaney, and Derek Greene. 2016. Novel2Vec: Characterising 19th Century Fiction via Word Embeddings. In *Proceedings of the Irish Conference on Artificial Intelligence and Cognitive Science (AICS)*. CEUR Workshop Proceedings, Aachen, Germany, 1–12.

[17] Vivek Gupta, Ankit Saw, Pegah Nokhiz, Praneeth Netrapalli, Piyush Rai, and Partha Talukdar. 2020. P-SIF: Document Embeddings Using Partition Averaging. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, Vol. 34. AAAI, Palo Alto, CA, USA, 7863–7870. Issue 05.

[18] Birgit Hamp and Helmut Feldweg. 1997. GermaNet - A Lexical-Semantic Net for German. In *Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications*. ACL, Stroudsburg, PA, USA, 9–15. Issue W97-08.

[19] Verena Henrich and Erhard W Hinrichs. 2010. GernEdiT - The GermaNet Editing Tool.. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*. ACL, Stroudsburg, PA, USA, 19–24.

[20] Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-Strength Natural Language Processing in Python. https://doi.org/10.5281/zenodo.1212303

[21] Muhammad Khalifa and Aminul Islam. 2020. Will Your Forthcoming Book be Successful? Predicting Book Success with CNN and Readability Scores. *arXiv preprint* 2007.11073 (2020), 1–9.

[22] Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The Efficient Transformer. In *Proceedings of the International Conference on Learning Representations (ICLR)*. OpenReview.net, Amherst, MA, USA, 1–12.

[23] Vincent Labatut and Xavier Bost. 2019. Extraction and Analysis of Fictional Character Networks: A Survey. *Comput. Surveys* 52, 5 (2019), 1–40. https://doi.org/10.1145/3344548

[24] Guillaume Lample and Alexis Conneau. 2019. Cross-Lingual Language Model Pretraining. In *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*. Curran Associates, Inc., Red Hook, NY, USA, 7057–7067.

[25] Quoc Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents. In *Proceedings of the International Conference on Machine Learning (ICML)*. PMLR, Brookline, USA, 1188–1196.

[26] O-Joun Lee and Jason J. Jung. 2020. Story Embedding: Learning Distributed Representations of Stories based on Character Networks (Extended Abstract). In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*. AAAI, Palo Alto, CA, USA, 5070–5074. https://doi.org/10.24963/ijcai.2020/709

[27] Cheng Li, Mingyang Zhang, Michael Bendersky, Hongbo Deng, Donald Metzler, and Marc Najork. 2019. Multi-view Embedding-Based Synonyms for Email Search. In *Proceedings of the ACM SIGIR Conference on Information Retrieval (SIGIR)*. ACM, New York City, NY, USA, 575–584. https://doi.org/10.1145/3331184.3331250

[28] Jana Lippmann and Nora Bechler. 2019. Buchproduktion. https://www.boersenverein.de/markt-daten/marktforschung/wirtschaftszahlen/buchproduktion

[29] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint* 1907.11692 (2019), 1–10.

[30] Yong Luo, Jian Tang, Jun Yan, Chao Xu, and Zheng Chen. 2014. Pre-Trained Multi-View Word Embedding Using Two-Side Neural Network. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, Vol. 28. AAAI, Palo Alto, CA, USA, 1982–1988. Issue 1.

[31] Suraj Maharjan, John Arevalo, Manuel Montes, Fabio A. González, and Thamar Solorio. 2017. A Multi-Task Approach to Predict Likability of Books. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*. ACL, Stroudsburg, PA, USA, 1217–1227. https://doi.org/10.18653/v1/E17-1114

[32] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. In *Proceedings of the International Conference on Learning Representations (ICLR)*. OpenReview.net, Amherst, MA, USA, 1–12.

[33] George A. Miller. 1998. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA, USA.

[34] Arvind Neelakantan, Jeevan Shankar, Alexandre Passos, and Andrew McCallum. 2014. Efficient Non-Parametric Estimation of Multiple Embeddings per Word in Vector Space. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. ACL, Stroudsburg, PA, USA, 1059–1069. https://doi.org/10.3115/v1/D14-1113

[35] Maria Soledad Pera and Yiu-Kai Ng. 2013. What to Read Next? Making Personalized Book Recommendations for K-12 Users. In *Proceedings of the ACM Conference on Recommender Systems (RecSys)*. ACM, New York City, NY, USA, 113–120. https://doi.org/10.1145/2507157.2507181

[36] Jiezhong Qiu, Hao Ma, Omer Levy, Wen-tau Yih, Sinong Wang, and Jie Tang. 2020. Blockwise Self-Attention for Long Document Understanding. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. ACL, Stroudsburg, PA, USA, 2555–2565.

[37] Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC Workshop on New Challenges for NLP Frameworks*. ELRA, Valletta, Malta, 45–50.

[38] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. ACL, Stroudsburg, PA, USA, 3980–3990. https://doi.org/10.18653/v1/D19-1410

[39] Laura Rettig, Julien Audiffren, and Philippe Cudré-Mauroux. 2019. Fusing Vector Space Models for Domain-Specific Applications. In *Proceedings of the IEEE International Conference on Tools with Artificial Intelligence (ICTAI)*. IEEE, New York City, USA, 1110–1117. https://doi.org/10.1109/ICTAI.2019.00155

[40] Julian Risch, Samuele Garda, and Ralf Krestel. 2018. Book Recommendation Beyond the Usual Suspects. In *Proceedings of the International Conference on Asian Digital Libraries (ICADL)*. Springer-Verlag, Heidelberg, Germany, 227–239. https://doi.org/10.1007/978-3-030-04257-8_24

[41] Max Roser. 2013. Books. https://ourworldindata.org/books

[42] Lei Sang, Min Xu, ShengSheng Qian, and Xindong Wu. 2019. Multi-Modal Multi-View Bayesian Semantic Embedding for Community Question Answering. In *Neurocomputing*, Vol. 334. Elsevier, Amsterdam, Netherlands, 44–58. https://doi.org/10.1016/j.neucom.2018.12.067

[43] Jannik Strötgen and Michael Gertz. 2013. Multilingual and Cross-Domain Temporal Tagging. In *Language Resources and Evaluation*, Vol. 47. Springer-Verlag, Heidelberg, Germany, 269–298. Issue 2. https://doi.org/10.1007/s10579-012-9179-y

[44] Yla R. Tausczik and James W. Pennebaker. 2010. The Psychological Meaning of Words: LIWC and Computerized Text Analysis Methods. In *Journal of Language and Social Psychology*, Vol. 29. SAGE Publications, London, England, 24–54. Issue 1. https://doi.org/10.1177/0261927X09351676

[45] Narendra Babu Unnam and P. Krishna Reddy. 2020. A Document Representation Framework with Interpretable Features Using Pre-Trained Word Embeddings. *International Journal of Data Science and Analytics* 10, 1 (June 2020), 49–64. https://doi.org/10.1007/s41060-019-00200-5

[46] Andreas van Cranenburgh, Karina van Dalen-Oskam, and Joris van Zundert. 2019. Vector space explorations of literary language. *Language Resources and Evaluation* 53, 4 (2019), 625–650. https://doi.org/10.1007/s10579-018-09442-4

[47] Paula Cristina Vaz. 2012. LitRec vs. Movielens - A Comparative Study:. In *Proceedings of the International Conference on Knowledge Discovery and Information Retrieval (KDIR)*. SciTePress, Setúbal, Portugal, 370–373. https://doi.org/10.5220/0004145003700373

[48] Feng Wang, Lingling Zhang, and Xin Xu. 2020. A Literature Review and Classification of Book Recommendation Research. *Journal of Information System and Technology Management* 5 (March 2020), 15–34. https://doi.org/10.35631/JISTM.516002