Detecting Paraphrases of Standard Clause Titles in Insurance Contracts

Frieda Josi
University of Applied Sciences and Arts Hanover
frieda.josi@hs-hannover.de

Christian Wartena
University of Applied Sciences and Arts Hanover
christian.wartena@hs-hannover.de

Ulrich Heid
University of Hildesheim
Institute for Information Science and Natural Language Processing
heid@uni-hildesheim.de

Abstract

For the analysis of contract texts, validated model texts, such as model clauses, can be used to identify used contract clauses. This paper investigates how the similarity between titles of model clauses and headings extracted from contracts can be computed, and which similarity measure is most suitable for this. For the calculation of the similarities between title pairs we tested various variants of string similarity and token based similarity. We also compare two additional semantic similarity measures based on word embeddings using pre-trained embeddings and word embeddings trained on contract texts. The identification of the model clause title can be used as a starting point for the mapping of clauses found in contracts to verified clauses.

1 Introduction

The calculation of text similarities is a key factor in the analysis of texts that consist of recurring text parts or that have to correspond to formulation patterns. In the insurance industry, there is a multitude of individual contracts between companies and insurance companies, or between insurance companies and reinsurance companies. However, most contracts more or less standardized clauses and text templates. In order to find the structure of a contract and to support the contract review, it is important to find all parts in the contract that are based on standardized clauses.

In our work, we compare a heading in the contract to be analyzed with all clause titles in a collection of model clauses. We have two reasons to do this title-based comparison: in the first place we work with scanned PDF texts. Thus we have to reconstruct the often complicated layout structure of the contract text. For the extraction of text we apply *pdfminer*, a Python PDF parser [1]. We use a trained classifier to identify headers, footers, enumeration elements, headings, stamps and hand-written remarks. We describe our procedure of layout-based structure recognition and analysis in Josi and Wartena (2018). Once we can identify titles of model clauses, we know what type of formatting is used for clause titles, and we can split up the main part of the contract text into a list of clause text blocks. Second, comparing the text of the clause bodies with all possible candidate model clauses requires less effort, if our system identifies the model clause candidate(s) based on their titles. An overview of the work presented here and its place in the overall project workflow is given in Figure 1. The subject of this paper concerns the first point from Figure 1, the similarity calculation of the model clauses.

In some cases, the title found in a contract is identical to the title of a model clause. In many cases, however the titles differ. We can identify many patterns of variation, such as addition or omission of

[1] PDFMiner: https://pypi.org/project/pdfminer/

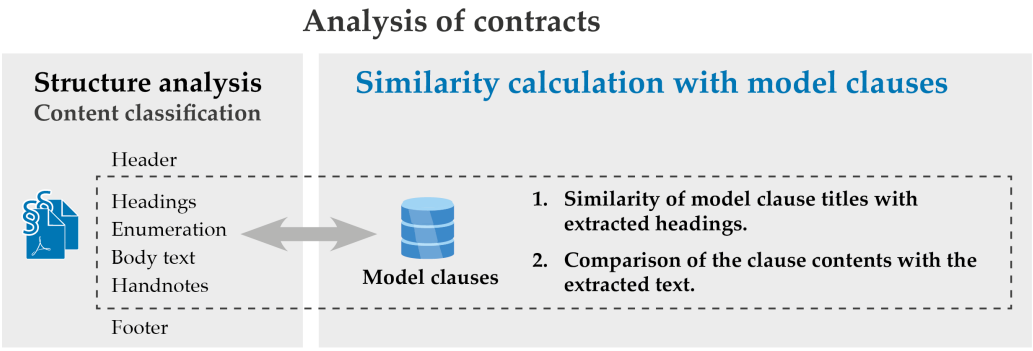


Figure 1: Overview of contract analysis with the section of similarity calculation of model clauses

Table 1: Titles of extracted clauses and their corresponding model clauses.

Extracted title	**Model clause title**	**Change**
ULTIMATE NET LOSS CLAUSE	Ultimate Net Loss	Extension
CONTRACT CONTINUITY CLAUSE (LSW 1035)	Contract Continuity Clause - Retro	Refinement
Choice of Law And Jurisdiction:	Governing Law and Jurisdiction	Lexical substitution
Arbitration ARIAS (UK) 1997 (G231.n:-	Arias Arbitration Clause	Refinement
Condition 15 ERRORS AND OMISSIONS	Errors and Omissions	Extension listing

the word *Clause*, addition of a colon at the end, a number in the beginning, etc. Some examples of such variations are given in Table 1. In addition, we have OCR errors and errors in the extraction of the headings, especially when the headings consist of several lines and are placed in the left margin, which is quite normal for insurance contracts.

If we manually defined a number of patterns of allowed variations, we would risk overfitting on the clauses we have seen and missing many unseen variations. Instead, we would like to use a simple similarity measure to compare the clause titles. The rest of the paper deals with the selection of the best similarity measure for this task. In order to evaluate similarity measures we constructed two sets of clause titles. The first set contains pairs of corresponding and non-corresponding titles. The task here is to predict whether two titles correspond or not. The second set contains a large number of extracted headings from contracts. The task is to predict which headings correspond to the title of a model clause.

2 Related Work

The calculation of text similarity is used in many applications and projects and is constantly extended and improved. To calculate similarities between very short text pairs and to match the correct titles of the model clauses with the extracted headings, we have used state-of-the-art methods and measurement approaches which we have adapted for our application and trained with our data set of contract texts. The particular problem we faced is that the text pairs are very short. On average, the titles consist of 24 characters and not a single title has more than nine words.

Bär et al. (2012) define a semantic textual similarity with character and word n-grams by a semantic vector comparison and a word similarity comparison based on lexical-semantic sources is performed using various similarity features. The authors apply a logarithmic-linear regression model and use *Explicit Semantic Analysis*, a vector based representation of text for semantic similarity measurements, to replace nouns. Further evaluations of similarity measurements of longer sentences are described by Achananuparp et al. (2008). Their evaluation measures are based on semantic equivalence, when the sentence pairs do not have the same surface shape. Whereby: sentences are similar, when they are paraphrases of each other, contain the same subject, or when one sentence is the super set of the other.

In (Bhagat and Hovy, 2013) similarities of paraphrases are defined and analyzed in terms of e.g. synonym substitution or part substitution. In total they describe 25 possible substitution types. Some of these types are also contained in our dataset of clause titles and model clause titles. Kovatchev et al. (2018) compare texts with different lengths on the similarity of their meaning. They attach much importance to detailed error analysis and have built a corpus in which paraphrases and negations are annotated. For the text pairs, the similarity is measured by paraphrase deductions in both texts. Another approach focusing on the analysis of paraphrases is described by Benikova and Zesch (2017). In this paper, different granulation levels of paraphrase sentences and annotated verb argument structures of paraphrases are compared for the similarity calculation. They distinguish between event paraphrase, lexical paraphrase, wording and inverse lexical paraphrase. In (Agirre et al., 2012) the semantic equivalence of two texts with paraphrase differences is measured. This is achieved by using common tokens in the sentence. For the vectors of the sentences the cosine similarity is calculated. The text pairs in their work consist of 51 words up to 126 words. Gomaa and Fahmy (2013) suggest to combine several similarity metrics to determine string-based, corpus-based, and knowledge-based similarities. For the string based approach they use the *Longest Common Sub String (LCS) algorithm* and various editing distance metrics like *Jaro*, *Damerau-Levenshtein* and *N-grams*. For the term-based similarity measurement *Manhattan Distance*, *Cosine similarity* and *Jaccard Distance*. Also Aldarmaki and Diab (2018) evaluate combined models for similarity measurement and evaluate the results of a logistic regression classifier by calculating the cosine similarity between two sentence vectors. Lan and Xu (2018) compare and evaluate seven LSTM-based methods for sentence pair modeling on eight commonly used datasets, such as Quora (Question Pairs Dataset), Twitter URL Corpus, and PIT-2015 (Paraphrase and Semantic Similarity in Twitter). For small datasets, they propose the method *Pairwise Word Interaction Model* (introduced in (He and Lin, 2016)).

Boom et al. (2015) recommend a combined method of word embedding and tf.idf weighting to calculate the similarity of text fragments (20 words per fragment). In this method, text parts with terms of a high tf.idf weighting are used. Kenter and de Rijke (2015) present a text similarity calculation, where they use the similarity of word vectors to derive semantic meta features, which in turn are used for training a supervised classifier. Kusner et al. (2015) present a distance measurement between text documents based on word embeddings and the dissimilarity of two probability distributions over words. *Word Mover's Distance* calculates the minimum vector distance of words from one document to words from another document.

3 Chosen Similarity Measures

To determine the optimal similarity calculation of text pairs we use character-based and token-based measurement methods. As can be seen in Table 1, the text fragments of our text pairs are very short, sometimes a single title consists of only one word. The longest title has nine words in total, the shortest title consists of a total of 5 characters. Hence, we compare the similarity measurement methods on character and token basis.

In addition, we determine the similarity of the title pairs with the support of word embeddings. We use the Word Mover's Distance measurement (Kusner et al., 2015), which combines word overlap with word similarities, thus considering substitution of words by semantically similar words. As another measurement, we have calculated the cosine distance of the average values of the word vectors to obtain an optimal threshold for separating the prediction of whether a title pair matches. For both methods using word embeddings we have used embeddings trained on a corpus of reinsurance contracts as well as embeddings from the pre-trained GoogleNews model[2].

3.1 Trigram Overlap

For the character-based similarity of the title pairs, we use the Jaccard coefficient of the set of all character trigrams that can be extracted from the titles. We do not use special symbols for the beginning and the end

[2]Used pre-trained model: GoogleNews-vectors-negative300.bin

of the string. This causes a slightly lower influence of changes at the beginning and the end of the string. We use N-grams based on the methodology of Markov (1913) and Shannon (1948) and the calculation of the Jaccard coefficient as described by Jaccard (1901).

3.2 Edit Distance

The edit distance between two strings is defined as the minimum number of edit operations necessary to change one string into the other one. We then use the edit distance to determine the number of edits in relation to the length of the strings. If $d(s,t)$ is the edit distance between titles s and t, we used $sim_d = 1 - \frac{d(s,t)}{\max(|s|,|t|)}$. Thus the value of sim_d is 0 if no alignment is possible and 1 if s and t are identical. We calculate the distance between the title pairs based on the minimal edit distance (Levenshtein, 1966).

3.3 Weighted Edit Distance

The weighted edit distance makes use of includes the following penalties: for substitution, insertion and deletion we use a penalty of 0.1 if a non-alphabetic character is involved, and a penalty of 1.0 otherwise.

3.4 Word Overlap

To predict the best threshold between equivalent and non-equivalent title pairs based on their tokens, we calculate the word overlap with the Jaccard coefficient, as we did for the trigram overlap in Section 3.1. We include only words in the comparison and completely disregard differences in punctuation in all token based methods. Since titles consist of only few words and they often are not completely identical, we also test a variant using stemming. We used the Lancaster Stemmer (Paice, 1990) because it reduces the words to a very short stem allowing to match different words with the same root. The results are clearly better than those obtained using lemmatization only. In Table 2 some examples of stemmed title pairs are shown.

Table 2: Comparison of some examples of tokenized title pairs reduced with Lancaster stemming for word overlap calculation

Tokenized title pairs	**Tokenized title pairs + Lancaster stemming**
[reinstatement clause] [reinstatements]	[**reinst** claus] [**reinst**]
[reinstatement provisions] [reinstatements]	[**reinst** provid] [**reinst**]
[reinstatement] [reinstatements]	**[reinst] [reinst]**
[terrorism **exclusion** clause] [terror war **exclusion**]	[**ter exclud** claus] [**ter** war **exclud**]

3.5 Vector Space Model

For the cosine similarity we again use stemmed titles. We experiment with two different vector weights: term frequency and tf.idf. We compute the idf weights on the set of all extracted headings and all titles of model clauses. Consequently, words like *clause* get a low weight.

3.6 Average Word Embeddings

We use two similarity measures in which the words are represented by word embeddings. These methods should be able to detect the similarity of two sentences in which a word is replaced by a synonym. As a first approach we represent a title by the average of all word embeddings of the words in the sentence. We use the cosine distance to compare the representations of two titles. We use these averaged word embeddings both with pre-trained embeddings and with embeddings trained on texts from the insurance domain.

In order to train domain-specific word embeddings, we used a collection of 3,730 insurance contracts with 15,831,789 lemmatized words after removing stop words and punctuation. We first use our developed

layout-based structural analysis method to separate the contents of the contracts from the text elements in the headers and footers. For training word vectors we use the text from contract content. In addition, we use the text of the model clauses and some other full texts from contracts.

The texts of the insurance contracts that we use for training our model are pre-processed. We use the tokenization and sentence splitting of the Natural Language Toolkit (nltk)[3]. All texts are lemmatized by the TreeTagger (Schmid, 1994). Finally, we remove all stop words (stop word list from nltk). The Open-Source Toolkit gensim[4] is used for vector modelling. We train vectors with 150 dimensions and used a window size of 3.

3.7 Word Mover's Distance

Word Mover's Distance (WMD) is a measure that compares two probability distributions of words and is defined as the minimum effort that is needed to move the probabilities from words in the starting distribution to words in the other distribution. The effort is defined as the sum of the probabilities moving from each word to another word multiplied with their distance, where the distance between two words is defined as the Euclidian distance between the word embeddings of the words. In order to obtain a similarity measure between 0 and 1 based on the WMD, we define $sim_{\text{WMD}}(s,t) = \frac{1}{1+\text{WMD}(s,t)}$.

As weights for the word distribution we use again pure term frequency and tf.idf weighting, and again we test the method with Google News Word embeddings and with our own word embeddings.

4 Experimental Setup

In the first experiment we evaluate the various distance measures on a classification task in which equivalent pairs of titles have to be distinguished from non-equivalent pairs of titles. The second experiment is a retrieval experiment in which all titles corresponding to the title of a model clause have to be found in a set of all headings extracted from a number of contracts.

4.1 Classification of Equivalent and Non-Equivalent Title Pairs

To calculate the similarity, we use 309 model clauses provided by the insurance company. We use a small subset (3730 contracts) of a large number of available contracts for the development of analysis methods. The methods are successively tested on a more comprehensive set of contracts. For the selection of a similarity measure we took six contracts, extracted all headings and manually selected all model clauses used in these contracts. This resulted in a set of 103 pairs where the model titles and our headings match (our gold standard), and we built a negative test set with an incorrect assignment to a model clause title (103 negative title pairs).

To find the threshold that marks the separation between positive and a negative pairs, for each above mentioned similarity measure, we compute the accuracy with varying thresholds.

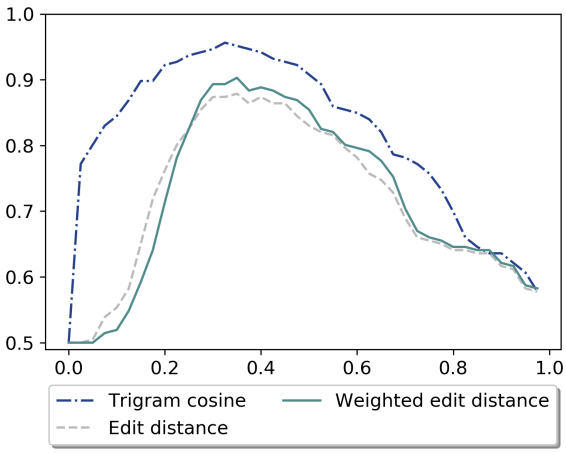
4.2 Retrieval of Model Clause Titles from a Set of Extracted Title Candidates

We analyzed six contracts and extracted all clauses with their headings from all clause text sections and manually annotated for each clause whether it corresponds to a model clause or not. The data set consists of 494 extracted headings for which a match is to be found in the data set of 154 model clause titles. The goal for the task at hand is to have an automatic method deciding for each heading, whether it is the heading of a clause corresponding to a model case or not. We can either define this as a binary classification problem or as a retrieval problem where we have to find all model clause headings from the set of all headings.

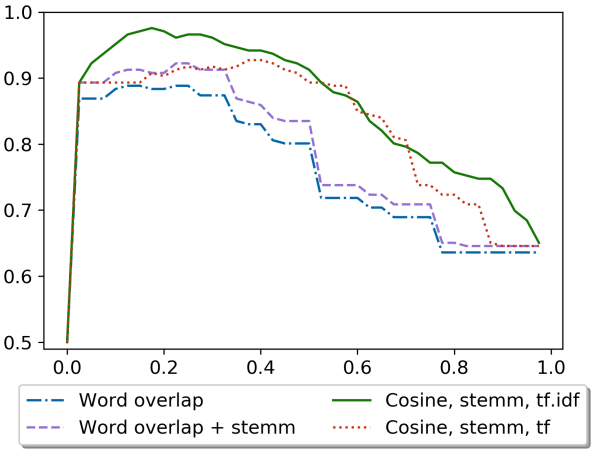
We use a kind of nearest-neighbor approach to solve the task: if the similarity of a title with some model clause title exceeds a defined threshold, we classify it as a model clause title. The goal of this

[3]Natural Language Toolkit: `https://www.nltk.org`

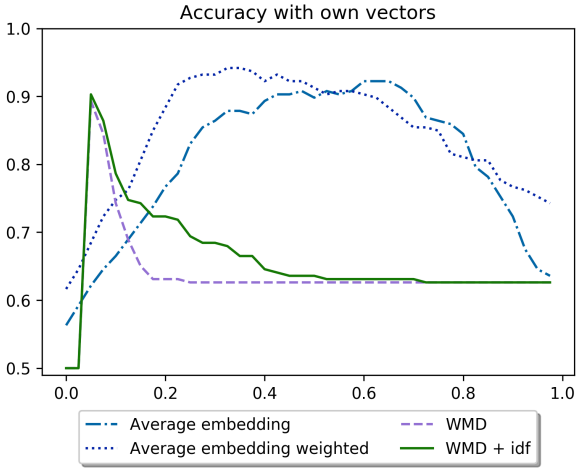
[4]Gensim library: `https://radimrehurek.com/gensim`



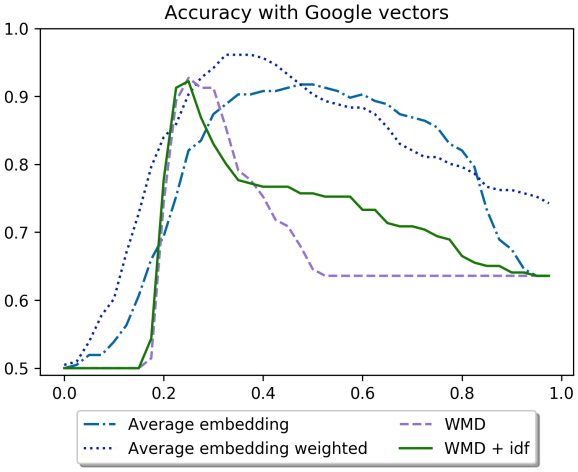
(a) Accuracy of character based similarity.



(b) Accuracy of token based similarity



(c) Accuracy of our word2vec model



(d) Accuracy of pre-trained word2vec model

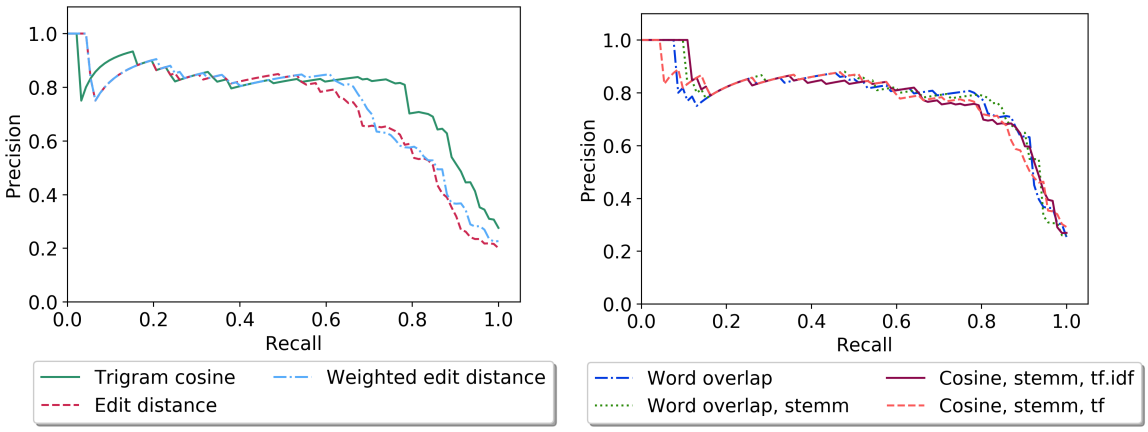
Figure 2: Accuracy of matching title pairs (Accuracy (y), Threshold (x)).

experiment is to find the subset of extracted titles which match the titles of model clauses. The similarity to the most similar model clause title gives a natural way to rank the result. We evaluate this ranking with the typical evaluation measures for rankings: average precision and area under the ROC curve (AUC). We also determine the maximum achievable accuracy and the corresponding optimal threshold to split the ranking into relevant and non-relevant results.

5 Results

The results of the first experiment (described in Section 4.1) are summarized in Table 3 and Figure 3. Here we identify the threshold for which we achieved the highest accuracy. This describes the similarity of the corresponding title pairs for the similarity measurement method we used. Table 5 shows examples and their prediction values for incorrectly predicted title pairs from the first experiment.

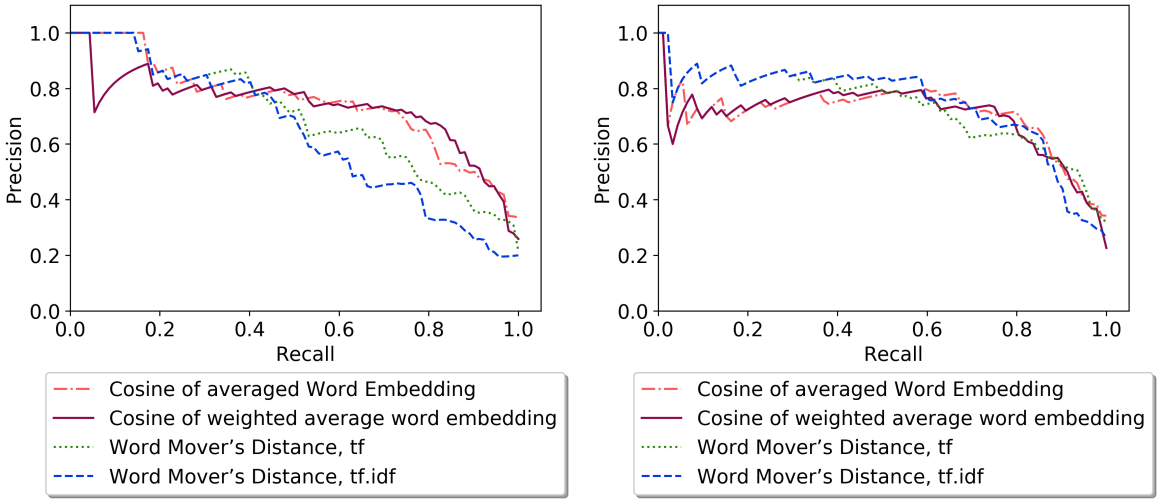
In Table 5, for the title pair [BIOLOGICAL OR CHEMICAL MATERIALS EXCLUSION][Genetically Modified Organism Exclusion Clause - Exclusion] only the measurement method *Cosine of weighted average word embedding* with pre-trained vectors of Google made a correct prediction (with 0.42). The pair contains different words that are semantically related. This kind of paraphrase is only captured when the word embeddings are used. However, in other cases these methods find a similarity where the titles do not correspond: In the case of the title pair [Class of Business:][Service of Suit] both methods using Google's



(a) String based methods

(b) Token based methods

Figure 3: Evaluate retrieval on decision of being a template heading



(a) Own Word Embeddings

(b) Google News Word Embeddings

Figure 4: Evaluate retrieval on decision of being a template heading, with Word Embeddings

pre-trained vectors made an incorrect decision. For title pairs like [Inspection of Records Clause][Access to Records] or [Choice of Law][Governing Law and Jurisdiction] the measurement methods with the pre-trained vectors and also the weighted cosine distance calculation made a correct prediction. The calculation of trigrams and word overlap give false negatives here.

The second experiment is described in Section 4.2. Table 4 and Figure 3 and 4 show the results for the methods that were used. Here we evaluated whether an extracted contract title corresponds to a model clause title. We show the threshold value we get for the highest accuracy. Then we calculate the average precision (AP) and the area under the curve (AUC).

6 Conclusion

The vector space model using cosine, stemming and tf.idf weights has achieved an accuracy of 0.98 in the classification task (experiment 1) and 0.91 in the retrieval task (experiment 2) and thus seems best suited to continue our work on contract analysis. The high accuracy of this method can be explained by the use of aggressive stemming, enabling the match of singular and plural forms of a word and by the use of idf weighting, which minimizes the influence of words like *clause, condition, retro* that are often added or

Table 3: Experiment 1: Classification of title pairs into corresponding vs. non-corresponding ones - max. accuracy

Method	Max accuracy	(Threshold)
String based		
Trigram cosine	0.96	(0.33)
Edit distance	0.88	(0.35)
Weighted edit distance	0.90	(0.35)
Token based		
Word overlap	0.89	(0.13)
Word overlap, stemming	0.92	(0.23)
Cosine, stemming, tf.idf	**0.98**	(0.18)
Cosine, stemming, tf	0.93	(0.38)
Custom Word Embeddings		
Cosine of averaged word embedding	0.92	(0.60)
Cosine of weighted average word embedding	0.94	(0.33)
Word Mover's Distance, tf	0.89	(0.05)
Word Mover's Distance, tf.idf	0.90	(0.05)
Google News Word Embeddings		
Cosine of averaged word embedding	0.92	(0.48)
Cosine of weighted average word embedding	0.96	(0.33)
Word Mover's Distance, tf	0.93	(0.25)
Word Mover's Distance, tf.idf	0.92	(0.25)

Table 4: Results for Retrieval-Evaluation (Experiment 2). Average precision (AP), Area under the curve (AUC)

Method	AP	AUC	Max Accuracy	(Threshold)
String based				
Trigram cosine	0.79	0.76	0.93	(0.60)
Edit distance	0.73	0.71	0.90	(0.59)
Weighted edit distance	0.74	0.73	0.91	(0.64)
Token based				
Word overlap	0.80	0.76	0.93	(0.40)
Word overlap, stemming	0.80	0.76	0.93	(0.40)
Cosine, stemming tf.idf	0.79	0.76	0.91	(0.60)
Cosine, stemming tf	0.79	0.76	0.92	(0.61)
Custom Word Embeddings				
Average word embedding	0.75	0.75	0.90	(0.80)
Cosine of weighted average word embedding	0.74	0.75	0.90	(0.68)
Word Mover's Distance, tf	0.70	0.72	0.88	(0.14)
Word Mover's Distance, tf.idf	0.64	0.67	0.87	(0.27)
Google News Word Embeddings				
Cosine of averaged word embedding	0.73	0.76	0.90	(0.76)
Cosine of weighted average word embedding	0.73	0.75	0.90	(0.66)
Word Mover's Distance, tf	0.76	0.76	0.89	(0.39)
Word Mover's Distance, tf.idf	0.77	0.75	0.90	(0.43)

Table 5: Incorrectly classified title pairs from experiment 1. Cells contain the computed similarity. In case the computed similarity leads to wrong classification (using the optimal threshold as given in the second line of the table), the cell has a red background.

Title pair	**Real**	**Trigram cosine**	**Word overlap, stemm.**	**Cosine, stemm., tf.idf**	**Google WMD, tf.idf**	**Google wgt. av. w2v**
Threshold		0.60	0.23	0.18	0.25	0.33
Inspection of Records Clause Access to Records	+	0.30	0.17	0.34	0.28	0.43
Choice of Law Governing Law and Jurisdiction	+	0.11	0.17	0.31	0.26	0.43
NOTICE OF LOSS Loss Settlements	+	0.15	0.25	0.28	0.22	0.17
Exclusions: Exclusions (general) - Exclusions	+	0.78	0.33	0.49	0.27	0.33
Simultaneous Settlements Clause Loss Settlements	+	0.55	0.25	0.46	0.23	0.27
BIOLOGICAL OR CHEMICAL MATERIALS EXCLUSION Genetically Modified Organism Exclusion Clause - Exclusion	+	0.36	0.1	0.07	0.22	0.42
CURRENCY CLAUSE Currency Conversion	+	0.54	0.33	0.62	0.29	0.72
INTERMEDIARY CLAUSE Period Clause	-	0.37	0.33	0.16	0.21	0.07
Class of Business: Service of Suit	-	0.14	0.2	0.07	0.25	0.33
TAXES Federal Excise Tax Clause	-	0.12	0.25	0.44	0.25	0.66
ULTIMATE NET LOSS Loss Settlements	-	0.14	0.25	0.24	0.23	0.23

removed from the standard title. Somewhat surprisingly, the WMD method does not give an equally good result. We attribute this to the fact that the exchange of synonyms rarely occurs in the title pairs. Thus, we also do not expect better results from other approaches using word embeddings.

Summarizing the result, we can conclude that there are no large differences between the different measures. In almost all cases, idf-weighting (using document frequency in headings) improves the results. Also, all methods using word embeddings yielded better results with the pre-trained Google News word embeddings than with the embeddings trained on our contract corpus.

Returning to our analysis of scanned PDF files: we need to find a significant number of model clause titles to find out what type of formatting is used for clause titles. Once we have detected the formatting of clause titles in a contract we can split up the contract into clauses, sub-clauses and so on and compare the full text of each clause with the model clause text. Experiment 2 shows that we can retrieve about half of the model clause titles with a precision of over 0.8 (see Figure 4). Interestingly, this high precision combined with 50% recall is only reached by the Word Mover's Distance with tf.idf values.

Acknowledgments

This research was financed by *Hannover Rück SE*. We would like to thank Dr. Julia Perl for many useful comments and *Hannover Rück SE* for making available the contracts.

References

Achananuparp, P., X. Hu, and X. Shen (2008). The Evaluation of Sentence Similarity Measures. In I.-Y. Song, J. Eder, and T. M. Nguyen (Eds.), *Data Warehousing and Knowledge Discovery*, Volume 5182, pp. 305–316. Berlin, Heidelberg: Springer Berlin Heidelberg.

Agirre, E., D. Cer, M. Diab, and A. Gonzalez-Agirre (2012). SemEval-2012 Task 6: A Pilot on Semantic Textual Similarity.

Aldarmaki, H. and M. Diab (2018). Evaluation of Unsupervised Compositional Representations.

Benikova, D. and T. Zesch (2017, November). Same same, but different: Compositionality of Paraphrase Granularity Levels. In *RANLP 2017 - Recent Advances in Natural Language Processing Meet Deep Learning*, pp. 90–96. Incoma Ltd. Shoumen, Bulgaria.

Bhagat, R. and E. Hovy (2013, September). What Is a Paraphrase? *Computational Linguistics 39*(3), 463–472.

Boom, C. D., S. V. Canneyt, S. Bohez, T. Demeester, and B. Dhoedt (2015, November). Learning Semantic Similarity for Very Short Texts. In *2015 IEEE International Conference on Data Mining Workshop (ICDMW)*, pp. 1229–1234.

Bär, D., C. Biemann, I. Gurevych, and T. Zesch (2012). UKP: Computing Semantic Textual Similarity by Combining Multiple Content Similarity Measures.

Gomaa, W. H. and A. A. Fahmy (2013, April). A Survey of Text Similarity Approaches. *International Journal of Computer Applications 68*(13), 13–18.

He, H. and J. J. Lin (2016). Pairwise Word Interaction Modeling with Deep Neural Networks for Semantic Similarity Measurement. In *HLT-NAACL*, pp. 937–948.

Jaccard, P. (1901). Étude comparative de la distribution florale dans une portion des Alpes et des Jura. *Bulletin del la Société Vaudoise des Sciences Naturelles 37*, 547–579.

Josi, F. and C. Wartena (2018). Structural Analysis of Contract Renewals. In *Proceedings of the ACM CIKM 2018 Workshops*, Turin.

Kenter, T. and M. de Rijke (2015). Short Text Similarity with Word Embeddings. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management*, CIKM '15, New York, NY, USA, pp. 1411–1420. ACM. event-place: Melbourne, Australia.

Kovatchev, V., M. A. Marti, and M. Salamo (2018). ETPC - A Paraphrase Identification Corpus Annotated with Extended Paraphrase Typology and Negation.

Kusner, M. J., Y. Sun, N. I. Kolkin, and K. Q. Weinberger (2015). From Word Embeddings to Document Distances. In *Proceedings of the 32d International Conference on Machine Learning - Volume 37*, ICML'15, pp. 957–966. JMLR.org. event-place: Lille, France.

Lan, W. and W. Xu (2018). Neural Network Models for Paraphrase Identification, Semantic Textual Similarity, Natural Language Inference, and Question Answering. arXiv: 1806.04330.

Levenshtein, V. I. (1966). Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics Doklady 10*.

Markov, A. A. (1913). Essai d'une recherche statistique sur le texte du roman "Eugene Onegin" illustrant la liaison des epreuve en chain ('Example of a statistical investigation of the text of "Eugene Onegin" illustrating the dependence between samples in chain') | BibSonomy. pp. 153–162.

Paice, C. D. (1990). Another Stemmer. *SIGIR Forum 24*(3), 56–61.

Schmid, H. (1994). Probabilistic Part-of-Speech Tagging Using Decision Trees. In *International Conference on New Methods in Language Processing*, Manchester, UK, pp. 44–49.

Shannon, C. E. (1948). A Mathematical Theory of Communication. *Bell System Technical Journal 27*(3), 379–423.