Detecting Scenes in Fiction Using the Embedding Delta Signal

Felix Schneider and Björn Barz and Joachim Denzler
Computer Vision Group
Friedrich Schiller Universität Jena
Jena, Germany
{felix.schneider, bjoern.barz, joachim.denzler}@uni-jena.de

Abstract

In this work we present a new method applied to a novel task: *Scene Segmentation*. This work is done in the context of the *Shared Task on Scene Segmentation (STSS)*. We propose the *Embedding Delta signal* as a novel method for both scene segmentation and topic segmentation. This delta signal represents the strength of the current change in context at any given point in a document. It is computed with a sliding window method, comparing cluster assignments of word embeddings in both halves of the sliding window using the cosine distance. Scene changes are found by searching for local maxima in the signal. We determine the type of the scene with a simple SVM approach. Hyperparameter search and SVM training is done on the 20 annotated German dime novels provided by the STSS organizers. The approach is then evaluated using both the per-sentence F1 score from the official STSS evaluation as well as the intersection over union of predicted and ground truth scenes. While showing low F1 scores of 0.02 and 0.04 for the tracks, we report an overlap of detected and ground truth scenes of 38% in both tracks.

1 Introduction

Narrative texts can be divided into different scenes. This task, called scene segmentation, is useful for analyzing narrative texts. In addition to the existing task of topic segmentation, scenes are internally consistent not only with the topic or action; also the story time and the discourse time during a scene are similar, additionally the space and the character constellation are internally consistent (Zehe et al., 2021a). While there exist approaches for the related field of topic segmentation, scene segmentation is a novel task. In this work, we present a method to approach this task in the scope of the *Shared Task on Scene Segmentation (STSS)* (Zehe et al., 2021b).

In addition to the direct benefits of automatically detecting scenes, which is of use for the analysis of texts, we see the search for anomalies in a narrative text as another application for scene segmentation. Since anomalies can be seen as a deviation from a homogenous context, the division of a narrative text into scenes can be used to provide internally homogeneous parts for the anomaly detection. One example for such an anomaly detection method is the MDI algorithm (Barz et al., 2019), which is able to detect anomalous intervals which deviate from the rest of the data in a given time series.

We present a method developed to provide a text segmentation and a simple kernel SVM approach to classify these segments as scenes or non-scenes. The segmentation method is a signal which provides a numerical value that represents the strength of the context change at any given sentence. It is inspired by the ideas of Burrows' Delta as well as topic segmentation methods like TopicTiling and TextTiling. We apply this method to the novel task of scene segmentation. The training data provided by the STSS organizers consists of 20 annotated German dime novels. An overview of the method is given in Figure 1.

2 Related Work

Scene segmentation is a novel task (Zehe et al., 2021a). However, there exists literature on topic segmentation and other related stylometric approaches, some of which are discussed below.

The basic idea of this work goes back to *Burrow's Delta* (Burrows, 2002), where the authors use frequency histograms of the most common words in a corpus to compare documents. This approach is common in the field of authorship attribution, since it measures the over- and under-usage of certain common words in a given document. However, since we want to measure the over- and under-usage

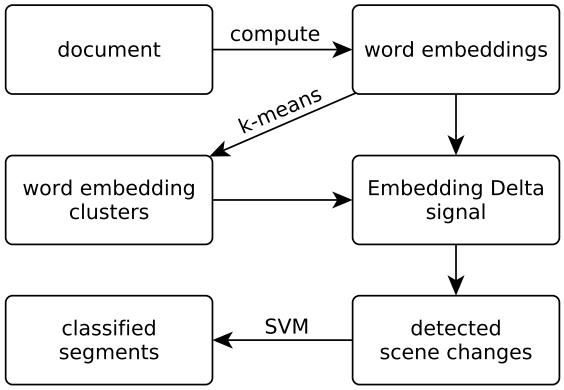

Figure 1: Overview of the document segmenting process.

not of certain words but instead of certain concepts, we compare the cluster assignment histograms of the word embeddings of a document instead. The most promising distance function for the Burrows' Delta is the cosine distance (Evert et al., 2017).

Another influence for our work was topic segmentation, which is closely related to topic modelling. A basic algorithm for topic segmentation is *TextTiling* (Hearst, 1997), which is used to segment a text into different pieces. Since term repetition is a strong indicator of topic cohesion (Halliday and Hasan, 1976), this algorithm compares adjacent paragraphs based on repetitions of contained words. The similarity score is computed based on the cosine distance between frequencies of previously detected items. In this, it shows similarities to Burrows' Delta.

In contrast to the more basic TextTiling, *TopicTiling* makes use of an LDA topic modelling approach (Riedl and Biemann, 2012). This method computes a coherence score between two adjacent sentences, given two windows containing a number of sentences of a fixed window size before and after the given sentence transition. The score is computed based on the cosine distance between the frequencies of the topics in the two windows.

A more modern approach is to incorporate word embeddings like *Word2vec* (Mikolov et al., 2013) into the topic modelling. Esposito et al. (2016) show that the use of word embeddings can improve the topic modelling capabilities of a system.

3 System Description

In this section, we describe how our approach to scene segmentation works. We propose a system to find change points inside of a text. Change points are points which divide a signal or other data in a way, such that the data before the change point differs from the data after the change point according to a defined criterion. In our task, we want to find points where the frequency of different concepts differs before and after the point. To achieve this, we move a sliding window over the text and compute the distance between representations of the first and of the second half of the window. In our application the window is defined for every point between two sentences in the text as a fixed number of sentences before and after this point. For every point between two sentences a numerical value is computed, which results in a signal indicating the strength of the change at any given point. We then search for peaks in this signal as those indicate the points with the strongest change. Peaks in our application means prominent local maxima in the signal. These points should be located between two scenes with different content. Afterwards, we use an SVM to distinguish the proposed parts between the peaks between scenes and non-scenes.

3.1 Embedding Delta Signal

The idea for the Embedding Delta signal is inspired by methods like the aforementioned Burrows' Delta, TextTiling, and TopicTiling. Instead of histograms of frequent words we use a histogram of word embedding cluster assignments to provide the vectors for the Delta measure. A histogram in our case is a vector, where each element of the vector represents the number of cluster assignments for word embeddings occuring in a certain part of a text.

The first step to create an Embedding Delta signal for a document is to compute the word embeddings (Bojanowski et al., 2017) for every word in the document. Then we use the k-means algorithm (Lloyd, 1982) to find clusters in the word embeddings of the document. For normalization in a later stage, we create a normalization vector v_{norm} by computing the cluster assignment histogram $h_{document}$ of the word embeddings of every word in the document, and normalize it by dividing it by its $L2$ norm, as shown in Equation 1.

$$v_{norm} = \frac{h_{document}}{||h_{document}||} \quad (1)$$

To generate the signal itself, we move a sliding window over the document, such that for every sentence i the window is centered on the beginning of

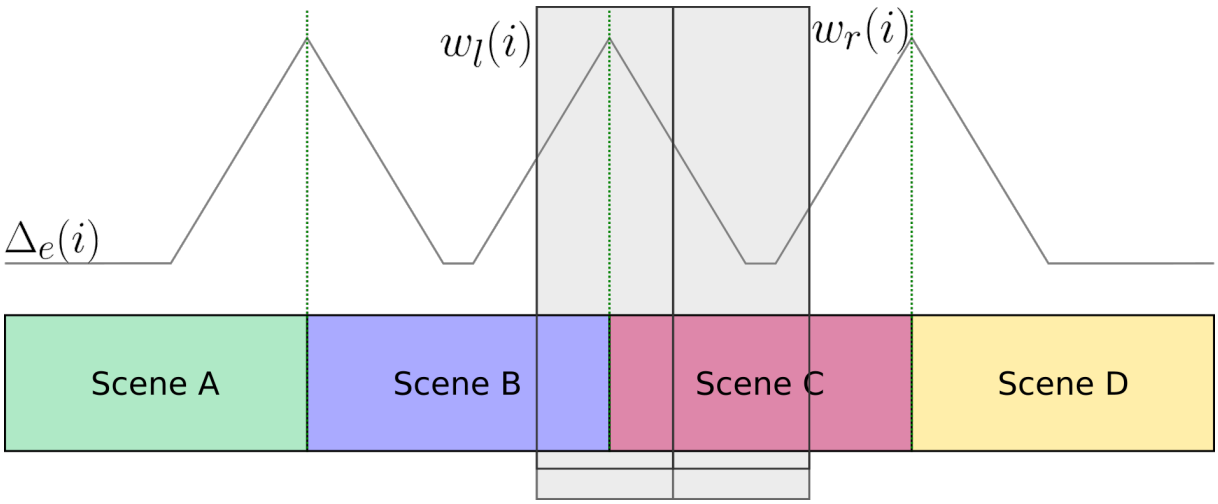

Figure 2: Sliding window generating the change signal.

the sentence. The sliding window contains a fixed number of sentences before and after the current position in its left and its right half. The number of sentences is a hyperparameter that has to be chosen for the application of the method. We then compute the histogram of cluster assignments v_h for the word embeddings of both the content of the first half and of the second half of the sliding window. In the next step, we divide both histogram vectors v_h by their $L2$ norm and substract the normalization vector as defined in Equation 2. The resulting vector is named w_l for the left side of the sliding window and w_r for the right side of the sliding window. In Equation 2 w can represent either w_l or w_r.

$$w = \frac{v_h}{||v_h||} - v_{norm} \tag{2}$$

We define the Embedding Delta signal Δ_e as the cosine distance between the normalized vectors of the first half and the second half of the sliding window as shown in Equation 3.

$$\Delta_e(i) = dist_{cos}(w_l(i), w_r(i)) \tag{3}$$

Consequently, peaks in the Embedding Delta signal Δ_e are considered scene changes. The peaks can be chosen by two different methods: The first way is to select all relative local maxima of the score, using a value called *order* to determine the number of points on either side of the potential local maximum that will be considered for comparison. The other way to select peaks is to use all local maxima whose value is greater than a certain threshold. The threshold for the second approach is $\mu + \frac{\sigma}{2}$ with μ being the mean and σ being the standard deviation of the signal, similar to TopicTiling and TextTiling.

3.2 Scene Type SVM

Given the change points from the previous step, we decide whether the part between two change points belongs to the *scene* category or to the *nonscene* category. For this, we train a Support Vector Machine (SVM) with an RBF kernel on the training data provided by the STSS. The features for the SVM comprise the mean number of characters of a sentence in a part, the standard deviation of the number of characters of the sentences, the number of characters in the whole part, and the number of sentences in the scene.

3.3 Additional Evaluation

The official STSS evaluation calculates the F1 scores for all correctly detected labels on a per-sentence basis: *Scene-Scene*, *Scene-Nonscene*, and *Nonscene-Scene*. This implies that a result where every part detection would be off by one sentence, the approach would have a F1 score of 0, even if most of the ground truth parts and detected parts overlap.

However, when scene segmentation is applied to the field of anomaly detection, also scene predictions are interesting, where the ground truth segments and the predicted segments mostly overlap, even if the borders are not exactly at the same place. To evaluate this, we also compute the *intersection over union (IoU)*.

For every ground truth part we find the detected part with the biggest overlap and assign it to the

ground truth part if it has not been assigned yet. We then add the length of all the overlapping regions and normalize them by the total length of the text, resulting in an intersection over union score value for the document. This score reaches its highest possible value when the ground truth parts and detected parts align perfectly. However, it does not take the distinction into scenes and non-scenes into account.

4 Experiments

In this section, we describe the exact experiments we carried out. We programmed them in python, using the packages numpy (Harris et al., 2020), scipy (Virtanen et al., 2020), scikit-learn (Pedregosa et al., 2011), and spaCy (Honnibal et al., 2020). The word vectors were obtained with the *de_core_news_lg* model from spaCy. The training and hyperparameter search was done on the 20 German annotated dime novels provided by the STSS organizers. We provide the source code[1] for the experiments.

4.1 Hyperparameter Search By F1 Score

Firstly, we searched for the parameters which resulted in the best F1 score for the STSS training data using the STSS evaluation script. For this, we tried window sizes of *15, 25, 35,* and *50* sentences for both of the window halves. Since previous evaluations showed the best results in this magnitude, we tried a number of *500* and *1000* clusters for the k-means model. The model was fitted with a maximum of 500 iterations to ensure convergence. The tested filter sizes for the smoothing were *5, 10, 20, 30, 40,* and *50*. As order for the search for relative maxima we tried *1, 10, 20, 30, 40,* and *50* points. In this step, all parts between detected peaks were considered as scenes.

Table 1 shows the hyperparameter configurations that resulted in the best F1 scores. The 15 mentioned configurations all had an F1 score of 0.02, while all other combinations had scores of 0.01 or 0. Since these results were non-conclusive, we conducted an additional experiment with a different evaluation approach.

4.2 Final Hyperparameter Selection By IoU

After the pre-selection of hyperparameters using the F1 score, the best hyperparameter set was chosen by computing the intersection over union. The bold line in Table 1 shows the hyperparameter set with the highest IoU of 0.42.

window size	clusters	kernel size	order
15	500	5	0
35	500	5	10
50	500	10	0
25	1000	10	0
25	**1000**	**5**	**20**
15	500	20	0
25	500	5	20
15	1000	10	0
50	500	5	20
25	1000	10	0
25	500	5	10
50	500	5	10
50	500	5	0
25	1000	10	10
35	500	5	0

Table 1: This table shows the best parameters from the hyperparameter search. They all result in an F1 score of 0.02. An order value of 0 represents the search non-relative maxima in the score.

Figure 3 shows an example Embedding Delta signal with the ground truth changes marked. We created this signal with the chosen hyperparameters from the previous step. It can be seen that the changes are in many cases at the peaks of the signal or close to them. However, for the per-sentence-evaluation of scene changes, the peaks must be at the exact locations of the ground truth scene changes. It can also be noticed that there are a few prominent peaks which do not have a corresponding ground truth scene change, and that some scene changes only have a relatively low score and even lie at local minima. If two adjacent text segments have very similar content and differ only in, e.g., the time, this method will produce a low Embedding Delta signal value and cannot distinguish between the segments.

4.3 SVM training

While the main part of our approach is the detection of change points and thus boundaries of scenes or scene-like parts in a document, we also wanted to approach the problem of distinguishing between the labels of *scene* and *non-scene*. As described above, we used a kernel SVM to detect the segment type. We used a 10-fold crossvalidation to determine the best C hyperparamter with the highest

[1] https://github.com/cvjena/embedding-delta

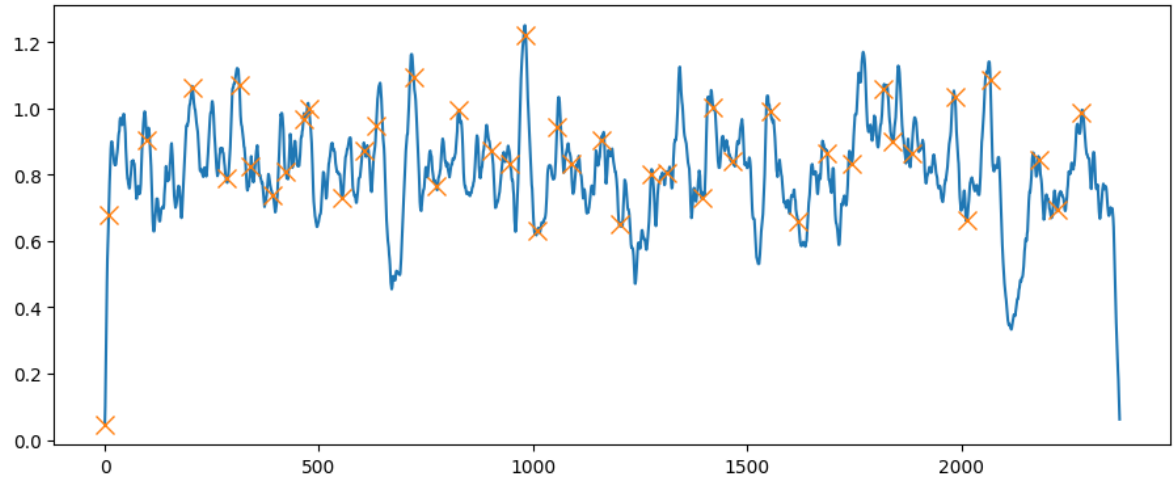

Figure 3: An example of the generated Embedding Delta signal. The scene changes are marked in yellow.

	micro avg. F1 score	IoU
track 1	0.02	0.38 ± 0.02
track 2	0.04	0.38 ± 0.01

Table 2: Results table for the test tracks. Intersection over union (IoU) is presented with the standard deviation over the documents.

F1 score. We tested *0.001, 0.1, 1, 10, 100, 1,000, 10,000, and 100,000* as values for regularization parameter *C* and they all resulted in an accuracy score of 0.96 or 0.97 for this binary classification task. We chose 100000 as value for C. While the score for this value was only 0.96, since all values had a similar score, we instead observed how well it could separate the classes when fitted and tested on the whole data. With a minimal worsening of *scene* accuracy from 1.0 to 0.997 for this parameter, an improvement in accuracy from 0.0 to 0.508 for the *non-scene* class was observed for this C value. The resulting SVM model was then used in the official evaluation.

4.4 Results

The official STSS evaluation was done on the described system. The system was tested on two different tasks: Task 1 consists of 4 annotated dime novels, task 2 consists of 2 annotated high literature texts. Additionally to the official evaluation, we computed the intersection over union for the provided results. Table 2 shows the results of the evaluation. While the F1 scores of both tracks differ, the IoU values are similar to each other.

5 Conclusion

In this work, we present an approach to the novel task of scene segmentation. The approach is influenced by methods from both the fields of authorship attribution as well as topic segmentation. We use a sliding window approach based on clusters of word embeddings to compute the cosine distance of the sentences surrounding a certain point. Thus, we generate a signal and from that a score to find scene changes. We can then group these found parts into scenes and non-scenes using a kernel SVM.

The results show on the one hand that the general approach is feasible to find the rough scene boundaries, as indicated by the high intersection over union. On the other hand, we see that the sentence-level F1 scores are still very low. This shows that even when the general position and lengths of parts of a documents are known, the exact locations of the boundaries are hard to find with our method.

However, the information that can be extracted with our method can still be useful for the analysis of narrative texts, as it holds data about the length and number of scenes in a text. Also for applications where only a rough knowledge of scene boundaries is important, the scenes detected by our method can be of use. One example for this can be the field of anomaly detection.

6 Further Work

There are multiple possibilities to further improve the approach described in this work. First, the creation of the two vectors in the sliding windows can be improved upon. One possibility would be to use fisher vector encoding (Sánchez et al., 2013)

instead of cluster assignments, which can include more information than a simple histogram.

Our current approach also does not use named entity recognition or similar methods. We have not found a useful way to incorporate this into our approach, but the presence of different named entities like persons or locations can on the one hand indicate a change point in the text, and on the other hand prove useful to determine whether a part is a scene or a non-scene. Another useful addition to the vectors could be features based on the verb tenses or the amount of direct speech.

Since our approach is based on a form of change point detection, also other methods from this field can be used. While we employ a sliding window approach, other change point detection methods like binary segmentation or bottom-up segmentation are also possible approaches (Truong et al., 2020). These search functions can be used with various cost functions besides the cosine distance, e.g., probability-based maximum likelihood estimations.

Finally, the main part of our approach - the Embedding Delta signal - is an unsupervised approach once hyperparameters have been chosen. However, the signal represents extracted information from the text which could in itself be used in an supervised model, like a conditional random field approach or another model from the field of machine learning/deep learning.

References

Björn Barz, Erik Rodner, Yanira Guanche Garcia, and Joachim Denzler. 2019. Detecting regions of maximal divergence for spatio-temporal anomaly detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1088–1101.

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

John Burrows. 2002. 'Delta': a Measure of Stylistic Difference and a Guide to Likely Authorship. *Literary and Linguistic Computing*, 17(3):267–287.

Fabrizio Esposito, Anna Corazza, and Francesco Cutugno. 2016. Topic modelling with word embeddings.

Stefan Evert, Thomas Proisl, Fotis Jannidis, Isabella Reger, Steffen Pielström, Christof Schöch, and Thorsten Vitt. 2017. Understanding and explaining delta measures for authorship attribution. *Digit. Scholarsh. Humanit.*, 32:ii4–ii16.

M. A. K. Halliday and R. Hasan. 1976. *Cohesion in English*. Longman, London.

Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. 2020. Array programming with NumPy. *Nature*, 585(7825):357–362.

Marti A. Hearst. 1997. Texttiling: Segmenting text into multi-paragraph subtopic passages. *Comput. Linguist.*, 23(1):33–64.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python.

S. Lloyd. 1982. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–137.

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Martin Riedl and Chris Biemann. 2012. Text segmentation with topic models. *JLCL*, 27.

Jorge Sánchez, Florent Perronnin, Thomas Mensink, and Jakob Verbeek. 2013. Image classification with the fisher vector: Theory and practice. *Int. J. Comput. Vision*, 105(3):222–245.

Charles Truong, Laurent Oudre, and Nicolas Vayatis. 2020. Selective review of offline change point detection methods. *Signal Processing*, 167:107299.

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. 2020. SciPy 1.0:

Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272.

Albin Zehe, Leonard Konle, Lea Katharina Dümpelmann, Evelyn Gius, Andreas Hotho, Fotis Jannidis, Lucas Kaufmann, Markus Krug, Frank Puppe, Nils Reiter, Annekea Schreiber, and Nathalie Wiedmer. 2021a. Detecting scenes in fiction: A new segmentation task. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3167–3177, Online. Association for Computational Linguistics.

Albin Zehe, Leonard Konle, Svenja Guhr, Lea Katharina Dümpelmann, Evelyn Gius, Andreas Hotho, Fotis Jannidis, Lucas Kaufmann, Markus Krug, Frank Puppe, Nils Reiter, and Annekea Schreiber. 2021b. Shared task on scene segmentation@konvens2021. In *Shared Task on Scene Segmentation*.