Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 137 (2018) 55–65

Procedia Computer Science

www.elsevier.com/locate/procedia

SEMANTiCS 2018 – 14th International Conference on Semantic Systems

# Semantic PDF Segmentation for Legacy Documents in Technical Documentation

Jan Oevermann[a,b,*]

[a] *University of Bremen, Bibliothekstraße 1, 28359 Bremen, Germany*
[b] *German Research Center for Artificial Intelligence (DFKI), Robert-Hooke-Straße 1, 28359 Bremen, Germany*

**Abstract**

The most common format to store and provide technical documentation is PDF. However, due to the unstructured nature of the format these documents are often excluded from a granular semantic access. While more and more companies are implementing XML-based component content management systems which can deliver annotated structured content, older legacy documents remain in their monolithic form.

We developed a new approach which segments PDF documents into semantically related sections via classification knowledge gained from structured training content. This approach based on machine learning is independent from any formatting information or visual clues.

In this paper, we take the results from multiple previous works and combine them into a holistic procedure model. We introduce a parameterizable range finding algorithm to refine segment detection and provide a RDF-based format to exchange the generated metadata which can then be used to improve information retrieval for users.



*Keywords:* PDF Segmentation; Technical Documentation; Machine Learning; Information Retrieval; Component Content Management

## 1. Introduction

In most regulated markets, such as the European Union, manufacturers are legally obliged to store and provide technical documentation for their products over a prolonged span of up to 30 years after bringing them into circulation [1]. To fulfill these regulatory requirements most companies store their content in form of PDF documents [12]. There are several reasons why the PDF format is suitable for this task, such as the focus on an exact reproduction of the content (especially the archival variant PDF/A), not allowing easy modifications and a widespread software support. At the same time manufacturers are faced with demands from their customers for a more sophisticated information

* Corresponding author
*E-mail address:* jan.oevermann@dfki.de

1877-0509 
10.1016/j.procs.2018.09.006

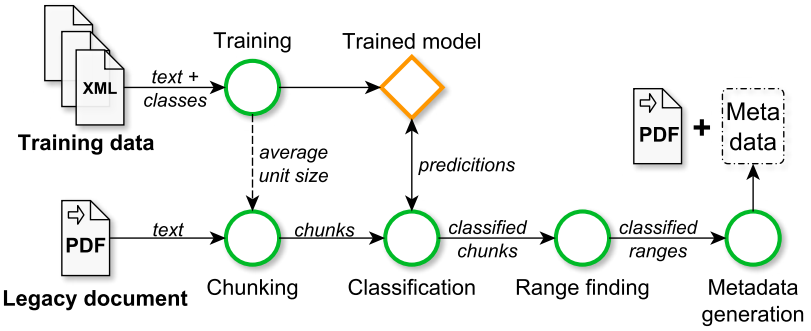


Fig. 1. Schematic data flow for segmentation process. A legacy PDF document is chunked and classified based on XML training data. Derived from the predictions a range finding algorithm can express semantic segments in standardized metadata format.

access, e.g. via content delivery portals (CDP) which include search and filtering functions. CDPs are constantly gaining popularity as primary method of information research for users [15].

While modern PDF versions include some functionality to provide a more structured access (e.g. tagged PDF) [2] these format variants are uncommon [6] or CDPs do not utilize the information due to technical restrictions. Some methods can split large PDF documents into multiple smaller ones[1] but this approach is often unsuitable for the sector of technical documentation because the once approved file cannot be modified in any way due to legal reasons. These aforementioned circumstances lead to the exclusion of legacy PDF documents from granular semantic access and restrict them to full text search or manual skimming.

The approach presented in this work leverages the fact that many companies use some form of XML-based component content management system to create, manage and publish their technical documentation [12]. In component content management text is written in a modular way which enables cost-efficient reuse and simple aggregation to compose documents. Content components[2] are often enriched with classifying metadata to describe certain semantic properties of the text (such as the information type or the described product component). This high-quality data can be used as training material in supervised learning and applied to the content from legacy PDFs to reconstruct semantically coherent segments.

To transfer the knowledge gained from classified content components to plain documents, the extracted text from a PDF is chunked into smaller units the model can be applied on. The underlying hypothesis states that chunks which contain overlaps of different semantic segments perform worse (in terms of classifier confidence) in the automated classification process. Based on per-chunk prediction and confidence results we can apply a range finding algorithm to reconstruct segments in the document. To use this generated metadata in a wide range of CDPs, we convert it to the recently introduced *intelligent information Request and Delivery Standard* (iiRDS) which is an exchange format for digital technical documentation.

iiRDS is developed by the *European Association for Technical Communication* (tekom) and combines a package format and a standardized RDF-based vocabulary to express corresponding metadata (which was partly derived from the PI classification method [11]). iiRDS has been released as a version 1.0 since April 2018 [4].

As we produce only additional metadata that can be delivered and processed alongside the original PDF file, we do not compromise any file-based approvals which is important in the regulated sector of technical documentation.

Our contribution beyond the state of the art is a dedicated procedure model for the semantic PDF segmentation of legacy document in technical documentation. Based on earlier work we extend the method with a profound range finding algorithm, a standardized metadata generation and a validation with real-world data. Furthermore we combine and conclude the previous approaches into a complete description of the procedure and a demo implementation.

[1] This kind of segmentation is often based on page ranges derived from the table of contents (TOC) or formatting information, such as headings, paragraphs or chapter breaks

[2] In other literature *content components* are also referred to as *topic*, *module*, *fragment* or *content module* [7]

## 2. Previous & Related Work

The method this work is based on was first introduced in [9], where the author presented the underlying hypothesis, a chunking algorithm and first tests with real-world data. In [3] the process was partially extended with a simple page range finding and metadata generation based on the WebAnnotation standard. The methodology for the automated classification of content components was extensively described and evaluated in [10].

In [5] the authors present a document segmentation technique which is based on probabilistic latent semantic analysis (PLSA). While their work is similar in assigning topics to different parts of a document based on "dips" in the similarity values of adjacent blocks, it is different in term of features used (PLSA vs *n*-grams), choice of "elementary blocks" (sentences vs content components) and the relevant measure for boundary detection (similarity vs confidence).

## 3. Methodology & Procedure

### *3.1. Prerequisites*

We define our *training data* as technical product documentation, provided as separate content components (between 50 and 500 words per unit) [10]. The training data must be annotated with *intrinsic* iiRDS concepts, meaning that they can be derived from text characteristics without additional information.

Examples for these concepts are instances of classes:

- `iirds:TopicType` (e.g. task, concept or reference)
- `iirds:InformationSubject` (e.g. technical data, safety information or process descriptions)
- `iirds:ProductLifeCyclePhase` (e.g. operation, maintenance or repair)
- `iirds:Component` (which serves as a docking point for company specific product components).

In most cases other established classification frameworks (such as PI-Class®[14] or eCl@ss[8]) can be directly mapped to these iiRDS concepts. The listed iiRDS concepts contain semantic information about the content itself which is why they are suitable for a semantic segmentation. Our goal is to generate metadata which defines semantically coherent segments in the PDF document defined as page ranges (e.g. "maintenance information can be found on pages 75–120").

As *test data* we define unstructured PDF documents with technical documentation which describes the same type of product from the same company as the training data does. Although our implementation relies on extractable text, the presented method can also work on text generated from an OCR preprocessing (e.g. for legacy documents which only exist as scans from paper copies). To evaluate the accuracy of the segmentation we define a baseline for each document which is based on the TOC[3] and manual assessment. This baseline is then compared to different segmentation approaches and parameter configurations.

### *3.2. Training*

The text from XML-based content components is extracted and sanitized. As features for training we chose a bag of n-grams model ($n = 2$) which weights n-grams according to the TC-ICF-CF method introduced in [10] to adjust for characteristics of component content management. For each class in the training data we build a prototypical vector with the normalized weighted features as its components. These vectors are used to create the trained model $M$ with all class vectors [10]. During the training phase we also store information about the average size of content components in words/unit: $a$.

[3] In technical documentation most TOCs follow a structure which resembles a combination of `iirds:InformationSubject` and `iirds:ProductLifeCyclePhase`

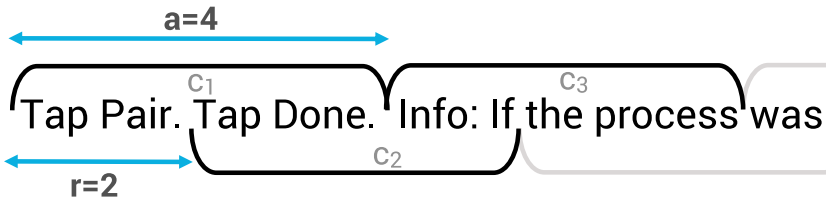


Fig. 2. Chunking example. $c_i$ is a generated chunk, Parameter $a$ is the size of a chunk as word count (typically between 50–200 words), Parameter $r$ is the offset with which chunks are generated as word count (typically a fraction of $a$, e.g. 40 words).

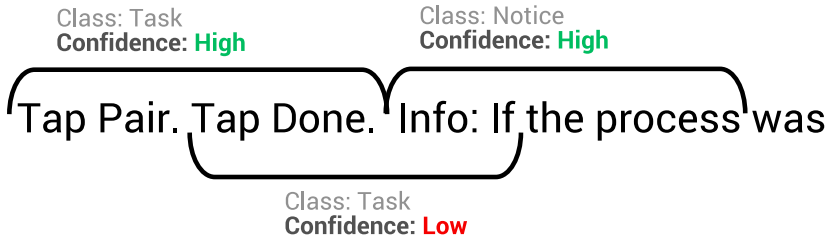


Fig. 3. Example for boundary between different text classes in combination with measured confidence for chunks. The text was extracted from a user manual, where a several steps of a task are followed by a information box.

### *3.3. Chunking*

The content from a PDF document is extracted as plain text. We discard any formatting information, since it is often either unreliable or unavailable for legacy documents (e.g. after OCR text extraction). As these "older" documents are the main focus of our work, we intentionally use an approach that only relies on plain text. The extracted text string is then split into words based on word boundaries (spaces, punctuation), which results in the set of extracted words $W$. We group the set of extracted words $W$ in arbitrary text chunks $C = \{c_1, ..., c_n\}$, where $c_i \subset W$. The size of chunks is based on the previously collected average word count of content components .

We chose $a$ as chunk size because it resembles the typical size (and, therefore, feature distribution) of an information unit for this specific product content which is crucial for a high classification accuracy. To distribute text chunks across the document content we choose a natural number $r$ as offset with $r \leq a$. This offset defines how multiple chunks overlap each other (cf. figure 2). Therefore, a text chunk $c_i$ at position $i$ can be defined as (for $i > 1$):

$$c_i = \{W_{(i-1)*r}, W_{(i-1)*r+1}, ..., W_{(i-1)*r+a}\} \tag{1}$$

The total number of chunks $|C|$ generated for a given set of words $W$ dependent on size and offset of chunks can be calculated as [9]:

$$|C| = \lfloor \frac{|W| - a}{r} \rfloor \tag{2}$$

A small value for $r$ increases the total number of chunks and therefore the resolution of boundary search, but has also a negative impact on classification performance. The offset can be chosen as a fraction of the average component size. We chose a default value of $r = \lfloor \frac{1}{4}a \rfloor$. Offsets smaller than this value do not offer significant advantages in interpreting the results while increasing computation time. For each generated text chunk we store content, size and position (relative to PDF pages, page number and total character count).

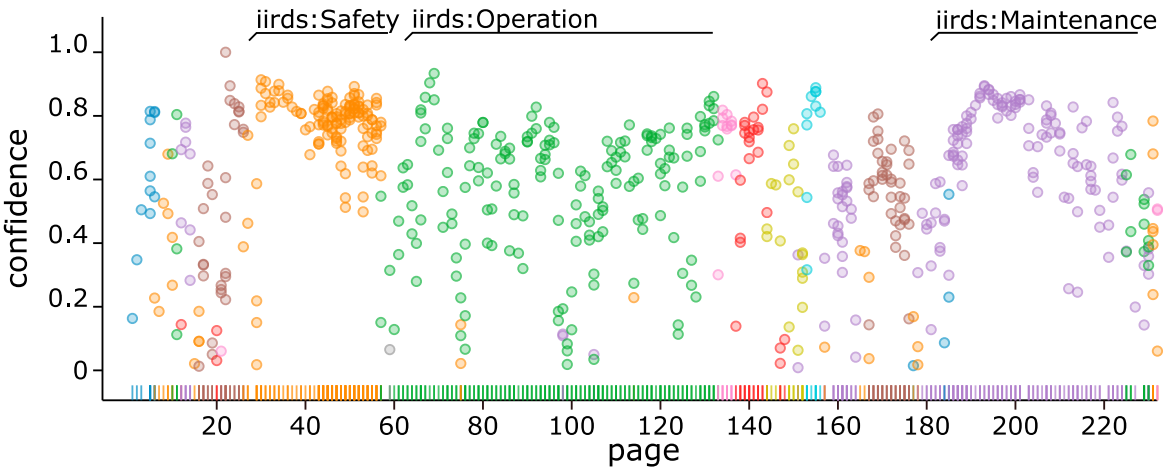


Fig. 4. Classification results plotted with confidence $p$ (y-axis), page position (x-axis) and predicted class (color). Three example segments that could be derived from this data were manually annotated. Outliers are clearly visible. Figure was reused from [3].

### 3.4. Classification & Confidence Scoring

All chunks in $C$ get classified by applying the feature extraction described in section 3.2 and calculating the cosine similarity for all class vectors in $M$ [10]. For the prediction (the class with the highest similarity $s$ in $n$ classes) an additional confidence score $p$ is calculated as

$$p = \frac{s_1 - s_2}{s_1 - s_n} \tag{3}$$

where per-class similarities are sorted from high ($s_1$) to low ($s_n$).

A typical distribution of confidence scores (y-axis) and predictions (color) across the pages of a document (x-axis) is shown in fig. 4.

The confidence score is based on the presence of single outliers (high confidence) or close runner-ups (low confidence) [9]. Predicted class and corresponding confidence score are stored alongside the chunk information. This confidence measure can be used to determine where boundaries between ranges of different classes are.

The underlying assumption for this is that when chunks are generated, some will contain (1) only text from one class while others (2) contain text from multiple classes. In case (1) the confidence score will be high because similarity is high for only one class. In case (2) multiple classes will have higher similarities and therefore a lower confidence score (cf. figure 3). These dips in the confidence curve across the document (cf. top section of figure 5) are indicators for boundaries between different segments.

### 3.5. Range finding

Previous range finding methods simply grouped text chunks with the same class into segments [3]. This approach is vulnerable to frequent outliers which break up larger segments into multiple smaller ones. With our approach we want to reconstruct macro structures like chapters or parts of the document which contain information about a specific part of the product.

The range finding method presented here is based on properties of the confidence curve and produces better results for real-world data. In our tests we could observe a significant improvement of range accuracy compared to the previous approach (cf. section 4). The range finding algorithm consists of the following steps (cf. figure 5):

1. Find all local minima ($p_{i-1} \geq p_i \leq p_{i+1}$) in the confidence values of the members of $C$, which are below a threshold $p_{minima}$ and put them into $N$
2. Cluster members of $N$ which are closer to each other than $i_{threshold}$ in $C$ into $N_{clustered}$

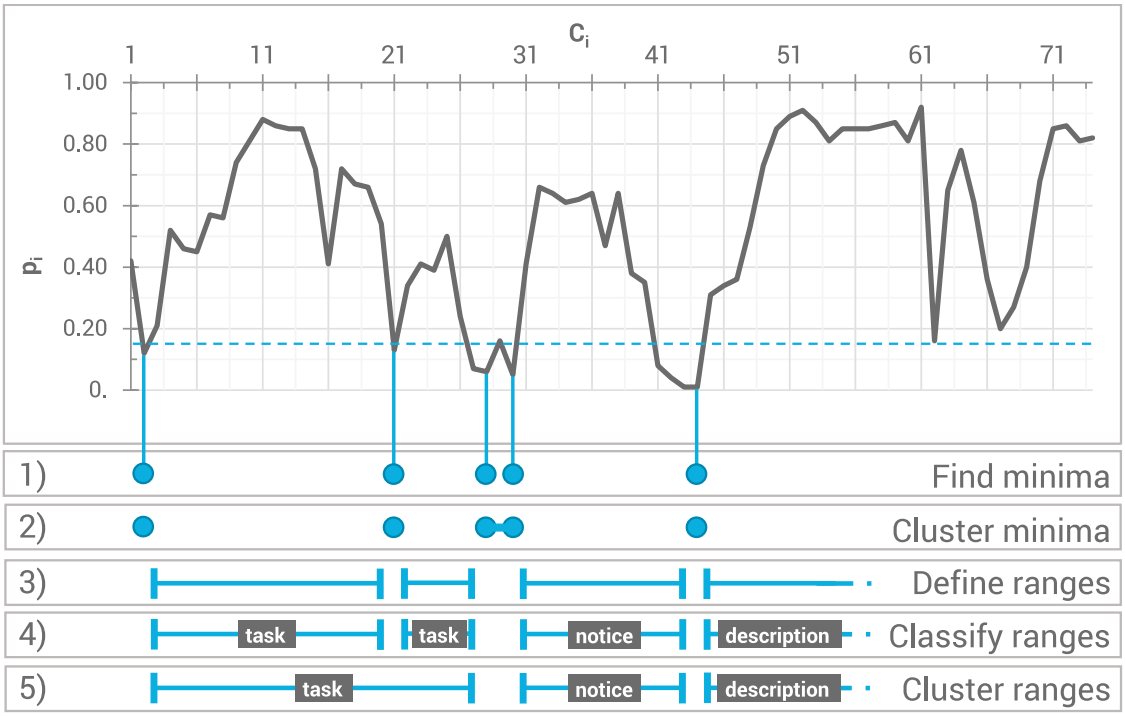


Fig. 5. Confidence curve based on a text excerpt of a PDF manual with derived range finding steps (cf. section 3.5). For this example we chose $p_{threshold} = 0.15$ and $i_{threshold} = 10$ as parameters. Chunking was done with $a = 172$ and $r = 42$ (both derived from the training data). The curve shows chunks 1–75 which cover pages 1–24.

3. Define ranges $R$ which contain all members of $C$ which are between $N_{clustered}$ are have a minimum length of $r_{threshold}$
4. Identify and label the predominant class for each member of $R$ by choosing the class with the highest median confidence (based on the confidence values of the predictions of the chunks which are within the examined $R_i$). Put the labeled members of $R$ which have a median confidence which are equal or higher than $p_{range}$ into $R_{labeled}$
5. Cluster members of $R_{labeled}$ with the same predominant class into $R_{clustered}$

With this multi-step approach the range finding process is parameterizable and can be adjusted to specific document-class-combinations. Due to the minima-based procedure we can define segments with a high average confidence, however, ambiguous parts of the document will not be allocated to a segment (as can be observed in figure 5 from chunks 27–30). This can be unfavorable for document splitting but can further improve precision in information retrieval tasks.

While the range finding algorithm is applied on chunk level (which usually is a resolution between 10–100 words) the results are converted to (full) page ranges because most CDP or PDF viewers do not provide other selector methods. An alternative to refine a defined range is to store the full text content of the identified segment alongside the metadata[4]. However, in most information retrieval use cases the range metadata is combined with some kind of text search which can point to a specific position within the segment.

### *3.6. Metadata generation*

To express the generated metadata for the PDF (page ranges and corresponding concepts) we can utilize the vocabulary provided by the iiRDS RDF schema. The standard is based on information units (`iirds:InformationUnit`)

[4] This can, for example, be achieved with an additional WebAnnotation `oa:TextQuoteSelector` as described in [3]

which act as an abstract class for the combination of metadata and content. Each information unit can have several physical renditions (`iirds:Rendition`) to provide the same content in different target formats. A rendition can directly point to a file or it can be refined by a selector logic to only select a specific range or position in the referenced file.

Therefore, we must generate one `iirds:Fragment` (which is the most suitable instance of an information unit) per range in $R_{clustered}$ with the predominant class in that segment as related metadata and a PDF rendition which is refined by a page range selector.

```
<iirds:Fragment rdf:about="urn:uuid:0b86fd8a-76b7-4cb9-ad41-2725edbf94c2">
  <iirds:has-subject
     rdf:resource="http://iirds.tekom.de/iirds#Safety"/>
  <iirds:has-rendition>
    <iirds:Rendition>
      <iirds:format>application/pdf</iirds:format>
      <iirds:source>files/manual.pdf</iirds:source>
      <iirds:has-selector>
          <iirds:RangeSelector>
              <iirds:has-start-selector>
                  <iirds:FragmentSelector>
                      <dcterms:conformsTo
                         rdf:resource="http://tools.ietf.org/rfc/rfc3778"/>
                      <rdf:value>page=15</rdf:value>
                  </iirds:FragmentSelector>
              </iirds:has-start-selector>
              <iirds:has-end-selector>
                  <iirds:FragmentSelector>
                      <dcterms:conformsTo
                         rdf:resource="http://tools.ietf.org/rfc/rfc3778"/>
                      <rdf:value>page=63</rdf:value>
                  </iirds:FragmentSelector>
              </iirds:has-end-selector>
          </iirds:RangeSelector>
      </iirds:has-selector>
    </iirds:Rendition>
  </iirds:has-rendition>
</iirds:Fragment>
```

Listing 1. Generated `iirds:Fragment` for example segment: Content about which has the (information) subject `iirds:Safety` can be found on pages 15 through 63 in the PDF document.

The set of generated fragments can then be compiled into an `iirds:Package` and delivered as an iiRDS ZIP container bundled with the associated PDF file.

## 4. Evaluation

For a first evaluation of our segmentation approach we use real-world technical documentation data provided from a company in the sector of heavy machinery. The XML-based training data consists of 663 content components with an average size ($a$) of 162 $\frac{words}{unit}$ in German. To each content component in the training set one of 10 iiRDS concepts was assigned by technical writers (in alphabetical order: `Assembly`, `EmergencyOperation`, `Formality`, `Maintenance`, `Operation`, `Safety`, `TechnicalData`, `TechnicalOverview`, `Transport`, `Troubleshooting`). In 10-fold cross validation this data set has a classification accuracy of 81.7%.

The PDF document we tested our method with is an unmodified manual with 234 pages of content about the same product group as the training data. The baseline structure (3.1) of the document is shown in table 1.

The simple range finding method introduced in [3] generates 59 ranges, which only partially overlap the baseline. As described in section 3.5 the high number of segments is caused by outliers in the classification. This high number of segments is unfavorable for information retrieval because it does not resemble the anticipated structure of technical documentation.

To evaluate our method we chose these parameters (with reference to section / step):

- chunking offset of $r = \lfloor \frac{1}{4} a \rfloor = 40$ (3.3)
- minima threshold $p_{minima} = 0.2$ (3.5 / 1)
- clustering threshold of $i_{threshold} = 5$ (3.5 / 2)
- range length threshold of $r_{threshold} = 5$ (3.5 / 3)
- confidence threshold of $p_{range} = 0.5$ (3.5 / 4)

These are the default values for the segmentation algorithm based on overall results for different documents. Better results for individual documents can be achieved through the adjustment of $p_{minima}$ dependent on the overall confidence distribution.

Table 1. Baseline structure of the analyzed document

| $R_i$ | Start | End | Concept |
|---|---|---|---|
| 1 | 1 | 8 | `iirds:Formality` |
| 2 | 9 | 14 | *TOC (not in training data)* |
| 3 | 15 | 28 | `iirds:TechnicalOverview` |
| 4 | 29 | 58 | `iirds:Safety` |
| 5 | 59 | 119 | `iirds:Operation` |
| 6 | 120 | 134 | *Equipment (not in training data)* |
| 8 | 135 | 138 | `iirds:Transport` |
| 9 | 139 | 146 | `iirds:EmergencyOperation` |
| 10 | 147 | 166 | `iirds:Troubleshooting` |
| 11 | 167 | 179 | `iirds:TechnicalData` |
| 12 | 180 | 228 | `iirds:Maintenance` |
| 13 | 229 | 234 | *Index (not in training data)* |

The advanced range finding algorithm presented here produces results which are close to the baseline segmentation (cf. table 2 and figure 6). Some of the deviations from the baseline can be explained by inaccuracies in recalculation of the page position for a given chunk which are caused by JavaScript character handling issues.

## 5. Implementation

We implemented the presented method as a client-side browser-based application written in JavaScript. The prototype is available online.[5] The source code is available in a public repository.[6] Due to the confidential nature of the content the example files cannot be provided.

The implementation is based upon earlier work [3]. We extended the software with a modified chunking, the new range finding algorithm (cf. section 3.5) and an iiRDS-conformant metadata output. The prototype can use XML and JSON data for training the model and PDF for analysis; results are output as RDF/XML following the iiRDS RDF schema.

[5] http://segments.fastclass.de
[6] https://github.com/j-oe/segments

Table 2. Results for semantic segmentation of the document

| $R_i$ | Start | End | Concept | $p(R_i)$ |
|---|---|---|---|---|
| 1 | 1 | 9 | `iirds:Formality` | 0,66 |
| 3 | 16 | 23 | `iirds:TechnicalOverview` | 0,83 |
| 4 | 24 | 55 | `iirds:Safety` | 0,83 |
| 5 | 62 | 126 | `iirds:Operation` | 0,67 |
| 8 | 126 | 145 | `iirds:Transport` | 0,81 |
| 9 | 145 | 151 | `iirds:EmergencyOperation` | 0,54 |
| 10 | 155 | 159 | `iirds:Troubleshooting` | 0,64 |
| 11 | 160 | 174 | `iirds:TechnicalData` | 0,80 |
| 12 | 174 | 225 | `iirds:Maintenance` | 0,75 |
| 14 | 231 | 232 | `iirds:Troubleshooting` | 0,70 |
| 15 | 232 | 234 | `iirds:Maintenance` | 0,74 |

## 6. Application

The industry use case for the presented method is metadata generation for legacy PDFs in technical documentation to improve access to the content. With the rising popularity of CDPs a structured and granular information retrieval becomes more important [15]. While XML-based content components can easily be provided in such a way, legacy documentation in PDF form is often excluded from these accessing methods [9]. Due to the use of iiRDS, the generated metadata can be used with any software which is compliant[7] with the standard. The segments can be utilized in faceted search to increase the general F-measure results in information retrieval [3]. These filtering methods allow the user to restrict a full text search to segments of different classifications (such as the type of information).

## 7. Outlook

The presented method could be applied to other document types (outside the realm of technical documentation) and alternative unstructured or weakly structured file formats (such as MS word files). Further evaluations with other segmentation techniques have to be carried out to continue to test our approach. A comparison or combination with the approach described in [5] would be of high interest.

## 8. Conclusion

We have developed a new approach to PDF segmentation which relies on semantic text properties and is, therefore, independent from formatting and visual representation of documents. Due to the characteristics of technical documentation we are able to leverage the knowledge in available structured data and apply it to unstructured documents to generate metadata which can be used in information retrieval applications. As shown with a first evaluation, we can reliably reconstruct segments of semantic importance with high accuracy in real-world data while minimizing the impact of outliers. Through the output of the results in a standardized RDF-based format, we can provide metadata to any iiRDS-compliant software to improve the information access to legacy technical documentation. For an easy validation of our method and as basis for future research we provide the implementation source code alongside a hosted prototype.

[7] the iiRDS selector logic is based on the W3C WebAnnotation standard[13]. Selectors were introduced to iiRDS with version 0.9 (RfC).

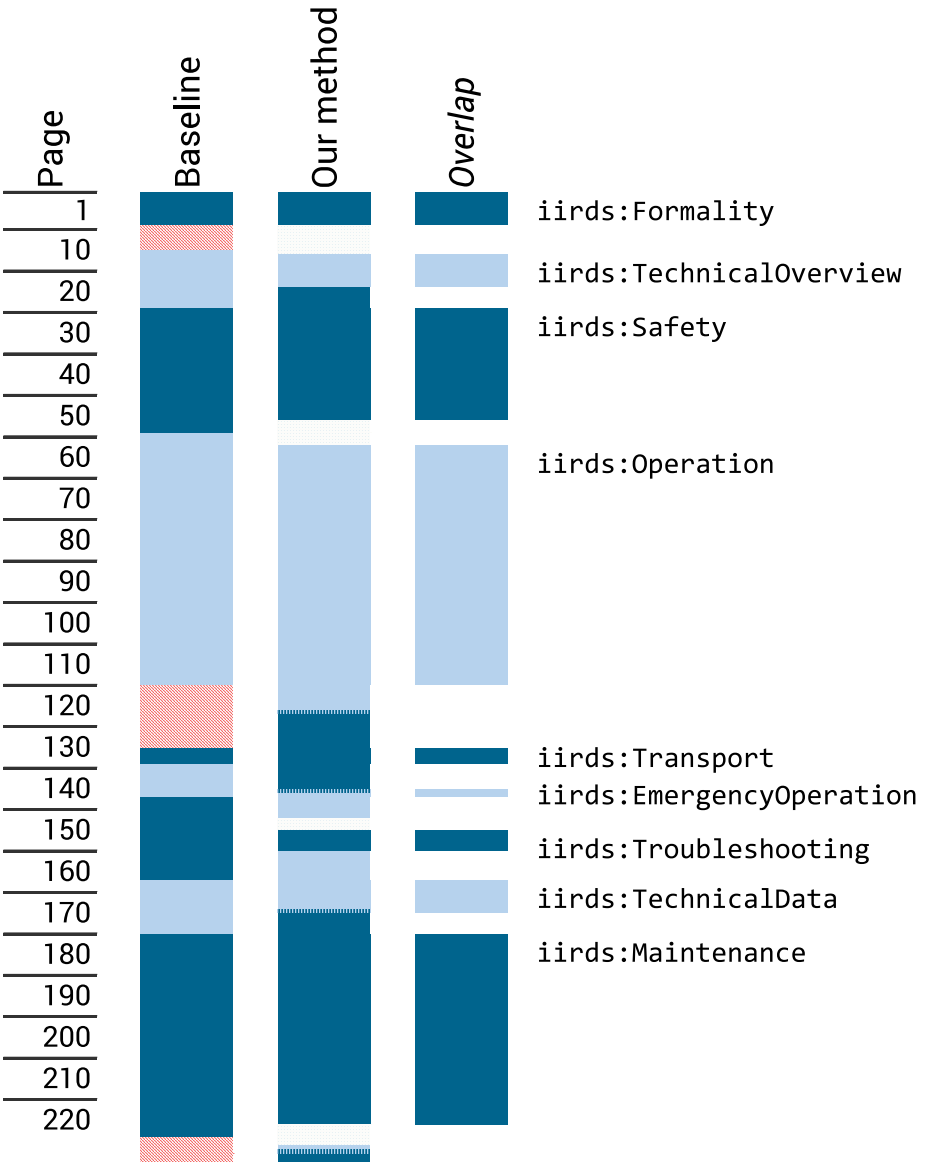


Fig. 6. Visualization of baseline segmentation vs our automated segmentation vs the overlap of both. Striped red segments were contents not available in the training data, dotted blue segments in our segmentation result are ranges with high uncertainty and, therefore, no classification. The overlap shows all page ranges which are in a correctly identified segment compared to the baseline.

**Acknowledgements**

The author wants to thank Christoph Lüth (University of Bremen) and Wolfgang Ziegler (Karlsruhe University of Applied Sciences) for insightful discussions and support.

This Research was supported by BMBF grant SELFIE, grant no. 01IW16001.

## References


[1] 2006/42/EC, 2006. Machinery directive of the European Parliament and of the Council.

[2] Adobe Systems (Ed.), 2001. PDF reference: Adobe portable document format version 1.4. Addison-Wesley, Boston.

[3] Bader, S., Oevermann, J., 2017. Semantic Annotation of Heterogeneous Data Sources: Towards an Integrated Information Framework for Service Technicians, in: Proceedings of the 13th International Conference on Semantic Systems. SEMANTiCS 2017, ACM, Amsterdam, The Netherlands. doi:10.1145/3132218.3132221.

[4] Becker, F., Kreutzer, M., Nuding, W., Oevermann, J., Parson, U., Sapara, J., Schubert, M., Steinacker, A., Wiedenmaier, M., 2018. iiRDS Specification - intelligent information Request and Delivery Standard - tekom Standard - 18 April 2018. URL: https://iirds.org.

[5] Brants, T., Chen, F., Tsochantaridis, I., 2002. Topic-based document segmentation with probabilistic latent semantic analysis, ACM Press. p. 211. doi:10.1145/584792.584829.

[6] Chao, H., Fan, J., 2004. Layout and Content Extraction for PDF Documents, in: Document Analysis Systems VI. Springer, Berlin, Heidelberg. volume 3163, pp. 213–224. doi:10.1007/978-3-540-28640-0_20.

[7] Drewer, P., Ziegler, W., 2014. Technische Dokumentation [eng.: Technical Documentation]. 2 ed., Vogel, Würzburg.

[8] Hepp, M., Leukel, J., Schmitz, V., 2007. A quantitative analysis of product categorization standards: content, coverage, and maintenance of eCl@ss, UNSPSC, eOTD, and the RosettaNet Technical Dictionary. Knowledge and Information Systems 13, 77–114. doi:10.1007/s10115-006-0054-2.

[9] Oevermann, J., 2016. Reconstructing Semantic Structures in Technical Documentation with Vector Space Classification, in: Proceedings of the Posters and Demos Track of the 12th International Conference on Semantic Systems. SEMANTiCS 2016, CEUR-WS, Leipzig, Germany. URL: http://ceur-ws.org/Vol-1695/paper5r1.pdf.

[10] Oevermann, J., Ziegler, W., 2018. Automated Classification of Content Components in Technical Communication. Computational Intelligence 34, 30–48. doi:10.1111/coin.12157.

[11] Parson, U., Sapara, J., Ziegler, W., 2017. iiRDS for Technical Writers - Introduction to the Metadata, in: Proceeding of tekom Conference 2017, tcworld, Stuttgart, Germany.

[12] Straub, D., 2016. Branchenkennzahlen für die Technische Dokumentation 2016 [eng.: Industry Figures for Technical Documentation 2016]. tcworld, Stuttgart, Germany.

[13] W3C, 2017. Web Annotation Vocabulary - W3c Recommendation 23 February 2017. URL: https://www.w3.org/TR/annotation-vocab/.

[14] Ziegler, W., 2015. Content Management und Content Delivery. Powered by PI-Class, in: Proceedings of tekom Conference 2015, tcworld, Stuttgart, Germany.

[15] Ziegler, W., Beier, H., 2015. Content delivery portals: The future of modular content. tcworld e-magazine .