Natural language processing methods for knowledge management - Applying document clustering for fast search and grouping of engineering

(article starts on next page)

Standard Article

Natural language processing methods for knowledge management—Applying document clustering for fast search and grouping of engineering documents

Ivar Örn Arnarsson[1] , Otto Frost[2], Emil Gustavsson[2],
Mats Jirstrand[2] and Johan Malmqvist[1]

Abstract

Product development companies collect data in form of Engineering Change Requests for logged design issues, tests, and product iterations. These documents are rich in unstructured data (e.g. free text). Previous research affirms that product developers find that current IT systems lack capabilities to accurately retrieve relevant documents with unstructured data. In this research, we demonstrate a method using Natural Language Processing and document clustering algorithms to find structurally or contextually related documents from databases containing Engineering Change Request documents. The aim is to radically decrease the time needed to effectively search for related engineering documents, organize search results, and create labeled clusters from these documents by utilizing Natural Language Processing algorithms. A domain knowledge expert at the case company evaluated the results and confirmed that the algorithms we applied managed to find relevant document clusters given the queries tested.

Keywords

natural language processing, document clustering, semantic data processing, knowledge management, engineering change request

Introduction

A product development (PD) project is a complex network of designs and design changes in which design attributes often have dependencies. Product developers often need to make changes to previous designs to improve, enhance, or adapt a product to identified opportunities. According to Eckert et al. (2000), when designers are confronted with a new design, their starting point is to look for old designs, often referring to the current design version or reusing other existing designs for components. Considering the pervasive trend of shorter development lead times, Prasad (2016b) argues that the search and re-use "information latency" needs to be brought down from days to minutes or even seconds, requiring more broadly applicable and faster information search tools. Moreover, the design will during the development process evolve

through a multitude of iterations, based on knowledge gained from simulations and tests, carried out within the current project as well as from other projects that a company is running in parallel. To efficiently iterate a design is a key capability in an interconnected concurrent engineering process (Prasad, 2016a).

The aim of this work is thus to automate to some extent designers' search and exploration so that time spent on this task can be drastically reduced and can

[1]Department of Industrial and Material Science, Chalmers University of
 Technology, Gothenburg, Sweden
[2]Fraunhofer-Chalmers Centre, Gothenburg, Sweden

Corresponding author:
Ivar Örn Arnarsson, Department of Industrial and Material Science,
Chalmers University of Technology, Chalmersplatsen 4, 412 96
Gothenburg, Sweden.
Email: varo@chalmers.se

instead be spent on analyzing and learning from search results. The idea is for designers to type in a search query about subjects they are interested in. The search engine employed in this project then performs a search and provides users with a list of related and relevant documents. This list of documents is then grouped into clusters so that the user gets an overview of the different topics the documents are related to. Solutions to problems are only of value if that content can be located quickly and accurately (Upshall, 2014); hence, we also developed a frontend application in which designers can submit their queries and view and navigate quickly through a user-friendly graphical user interface (GUI). It is often time consuming and difficult to find related documents in a list generated from a search into a large database. By providing clusters instead of long lists of documents, users can identify related documents faster and in a more structured fashion.

Information on product changes, including structured data (e.g. numerical data, categorical data, and timestamps) mixed with unstructured data (e.g. text inserted by engineers), is stored in databases to keep track of evolving design solutions, verification, and decision documents (Pikosz and Malmqvist, 1998). Large PD projects can contain tens of thousands of product change document, also known as Engineering Change Requests (ECRs) (Arnarsson et al., 2017; Jarratt et al., 2011). The ECR process focuses on achieving a rapid and quality-assured process from the identification of issues to the resolution. With the rapid development of data collection and storage, companies now confront large volumes of complex data from multiple sources (Wu et al., 2014). Around 80% of organizational data is in unstructured form (Grimes, 2008), and thus poses a challenge: how to extract meaning from this vast amount of text (Kobayashi et al., 2018). Arnarsson et al. (2017) affirm that product developers need to perform advanced searches within databases for efficient problem investigation and resolution, yet there are problems associated with this task: (i) search functionality is limited, mostly considering structured data and not taking into account the rich information stored in unstructured data (e.g. text data) and (ii) current search results return up to hundreds of reports even when filters are applied (e.g. timeframe, vehicle types, and keywords). Identifying the most relevant documents, therefore, becomes a time-consuming task.

It is important for product developers to be able to apply additional layers on top of search queries to further categorize current PD data. To identify structurally related information (e.g. all ECRs related to a particular part) and contextually related documents (e.g. ECRs dealing with a certain problem), a short list of the most relevant documents from both structured and unstructured data is produced. Recent developments in Natural Language Processing (NLP) have made it possible to search through huge databases of text documents to retrieve relevant items based on a variety of search queries. By utilizing NLP methods, such as search engine methods or document embedding approaches, we can find documents, in this case ECRs, that are similar by looking at the contexts and sequences of words together, instead of taking words in isolation. Previous methods are simpler; they rely on the notion that related documents share the exact same words, so to find similar documents, users usually search through the document set to find the ones that match best according to word count. More elaborate methods, such as doc2vec-methodology (Le and Mikolov, 2014), utilize the context of a document and automatically detect synonyms to words used in the search query.

Another benefit of these types of NLP methods is that they provide a similarity metric between documents, which means that users get a numeric value quantifying the degree of similarity between pairs of documents. This metric can be utilized for performing cluster analyses (Steinbach et al., 2000) where related documents into clusters are grouped together.

This paper explores the research gap in performing advanced searches based on NLP and clustering techniques within an ECR database. Such models can ensure better pre-studies for product developers as they can now evaluate a short list of the most relevant historical documents and their topics that might contain valuable knowledge. The document cluster analysis can be utilized for summarizing and grouping together documents with similar content, allowing more effective knowledge management of ECR documents.

The aim of this paper is to develop a method for and to test search queries that can be used for clustering ECR documents. The identication of relationships between documents support engineers in making better decisions for future projects. We specifically addressed two research questions:

1. *Can NLP and document clustering algorithms be utilized for grouping ECRs?*
2. *How do product developers benefit from automated document clustering?*

This paper is structured as follows: section "Earlier work" presents earlier work, the "Research approach" section outlines the research approach, section "Results" presents the results, section "Discussion" discusses the findings in relation to the research questions,

while section "Conclusions and future work" concludes the paper and discusses ideas for future research.

Earlier work

This section reviews earlier work on the data information needs of product developers, text analyses, NLP, and clustering algorithms. Arnarsson et al. (2017) interviewed 20 individuals with diverse experiences regarding the PD process to identify the information needs of product developers. They found that there is an opportunity to perform integrated searches across databases, including structured and unstructured data; however, this was not done in the case explored. Current IT systems have limited ability to search unstructured data, and the interviews revealed that most of the knowledge documented available is in the form of unstructured data, such as report title, description of changes, actions taken, comments, and technical solutions. Furthermore, the documents of interest should be correlated and clustered to help users identify related knowledge in various documents.

The ECR process is part of the engineering change process that corresponds to the change trigger approval states described by Jarratt et al. (2011) in their generic engineering change management process. The engineering change process takes place in different PD phases, and various authors have proposed generic models. Eckert et al. (2004) divided the sources of change throughout the design process into two categories: emergent changes and initiated changes. Jarratt et al. (2011) suggested a six-state engineering change process that begins with the ECR to be raised, the identification of solutions, a risk assessment of solution, the selection and approval of a solution, the implementation of the solution, and a review of the change.

The extraction of design and the manufacture of text content have been performed successfully using NLP and node models by, for example, Dong and Agogino (1997), who introduced a technique to induce a design representation of the design based on syntactic patterns found in the text corpus of design document; Catron and Ray (1991), who applied a node model to bridge requirement gaps between process planning and production control; and Kim and Wallace (2009), who presented a linguistically induced search approach applied on aerospace reports to improve the precision and recall of documents as current search engines only use traditional mathematical approaches. Dong (2005) further explored design team communication documents using a latent semantic approach on design documentation corpora and verbal communication to directly measure knowledge construction.

In more general terms, the development of NLP methods for summarizing and interpreting entire documents have increased over the past few years. Previous studies have extensively examined methods for transforming single words into high-dimensional representations, for example, Mikolov et al.'s (2013) word2vec. Le and Mikolov (2014) further developed this idea and introduced doc2vec, a word embedding methodology that trains a model to translate entire documents into a high-dimensional numerical representation. This numerical representation of documents can be utilized to compare different documents, find similar documents, and group documents into different themes. Two powerful properties that can be achieved by methods such as doc2vec include: (i) contextual information that can, in some cases, be interpreted and (ii) synonyms to words utilized in a document that can automatically be encoded in numerical representations.

Some classical clustering algorithms include k-means (Ahmad and Hashmi, 2016) that uses structured and unstructured datasets to find distance measures between data points and Newman's (2004) algorithm that detects and extracts community structure from networks based on the idea of modularity. Latent Dirichlet Allocation (LDA) is more tailored for text-based data and can be considered as an approach to analyze an underlying set of topics in text documents. LDA is a useful method for processing large collections of text and finding short descriptions that can be used for statistical relationships for tasks such as classification, summarization, and judgement of relevance and similarity (Blei et al., 2003). LDA is a generative statistical model, which is a form of unsupervised learning that views documents as bags of words (i.e. order does not matter) and then tries to find clusters that describe the different topics the documents seem to be about (Misra et al., 2008). LDA has been presented as a graphical model for topic discovery, allowing observations to be explained by unobserved groups; it is useful when dealing with large corpuses, and has been shown to outperform other dimension reduction techniques (Blei et al., 2003). Yoon et al. (2017) used LDA to examine new product opportunities by measuring the semantic similarities between patents and products, creating visual map portfolios that recommend untapped products. Prior studies have applied text clustering to optimize design structure matrices based on PD organizations (Yang et al., 2014) and PD project scheduling (Tripathy and Eppinger, 2013). Sarkar et al. (2014) applied spectral characterization to present a graph theoretic spectral approach that reveals hidden modular layers. Meanwhile, Yang et al. (2018) provided an innovative spectral clustering approach using similarities of team attributes and relationships based on PD organizational structure.

Table 1. Three steps of text mining models.

Step	Name	Description
1	Text pre-processing	Selecting, cleaning, and transforming documents into a suitable form (i.e. conversion of unstructured text into mathematical usable form)
2	Application of text mining operations	Applying algorithms for mining of patterns, clusters, association rule discovery, trend analysis, and other knowledge discovery algorithms
3	Post-processing	Manipulating the data from earlier steps to select, visualize, and validate knowledge

Other document clustering techniques include Non-negative Matrix factorization methods (NMF) and Latent Semantic Indexing (LSI) that also have shown promising results on large-scale text databases (Aggarwal et al., 2012).

Previous research identified three key components for conducting text mining, from Fayyad et al.'s (1996) work to newer work by Upshall (2014), which are in line with each other:

- Domain knowledge.
- Knowledge of the text-mining software and configuration.
- Knowledge of natural-language processing and computational linguistics.

Text mining models usually consists of three steps (Zhang et al., 2015; see Table 1).

Research approach

The objective of this work is to automate to some extent the exploration of information for designers so that they do not spend time searching for information to find a suitable source. The amount of stored data is growing, so it becomes harder to locate relevant information when designers perform search queries; they need additional layers to further sort information. The idea is to apply clustering algorithms on top of traditional searches when designers type in a search query on a subject they are interested in. Previous studies have not performed nor tested search engines in combination with clustering on ECR data.

The study is based on a PD database containing ECRs from real commercial vehicle PD projects within an organization. The NLP search method enables users to easily search through the data within the database using simple queries. We focus on unstructured data as previous findings suggest that current IT systems are deficient in searching and retrieving relevant documents based on text data. The methodology and the process used in this research start with the data from the ECR

database undergoing pre-processing in which stop word removal, lemmatization, cleaning, and concatenation are applied (see Table 2 for further description).

After pre-processing, data is fed parallel to the Elasticsearch (i.e. search engine) index and then used for the training of a doc2vec model. The resulting doc2vec model is indexed to enable fast retrieval of the document vectors. The doc2vec and Elasticsearch indexes are queried by the search service which contains the search and the clustering APIs. These APIs serve the search frontend application (see Figure 1).

The set-up of this research project as a partnership with the case company gave us access to a detailed inquiry about the topic in a real setting. This ability to access real-world data is the main rationale for selecting the single-company design (i.e. an opportunity to study a situation otherwise inaccessible to researchers), which Yin (2013) refers to as a "revelatory case."

The ECR data contains roughly 8,000 documents from recent development projects. Each ECR consists of more than 50 variables that are used to document and follow up on the logged changes. The documents are log files of design- and test-related issues that need to be resolved. They contain variables, such as title, part name, problem description, root cause, solution, and test results.

The search application considered in this project consists of three modules: the Elasticsearch, the doc2vec model, and the clustering application. These modules are linked by a frontend search service that can pass queries to both the Elasticsearch and doc2vec modules and summarize results using the clustering application.

For the doc2dec module (i.e. the model that translates each document into a high-dimensional numerical representation), a pipeline consisting of three steps was used. First, the data cleaning and normalization step reduced the cardinality of the set tokens in the dataset, reducing dataset complexity to improve the robustness and accuracy of the final model. This step includes lowercasing, removing non-letter characters, and lemmatizing using the WordNet lemmatizer (Bird and Loper, 2004). Second, the step trained a doc2vec model (Le and Mikolov, 2014) using the gensim library (Rehurek

Table 2. Pre-processing steps used in this research.

| Elements	Description
Stop word removal	Remove common terms to improve search accuracy. Example of stop words: "a," "the," "like," and "test."
Cleaning	Remove identified and redundant patterns. Remove malformatted words.
Lemmatization	Use WordNet to reduce words to their base form. Example: The verb "to walk" can also appear as "walk," "walked," "walks," and "walking." These words, therefore, all take on the base form "walk."
Concatenation	Combine the free text fields for doc2vec to a single field
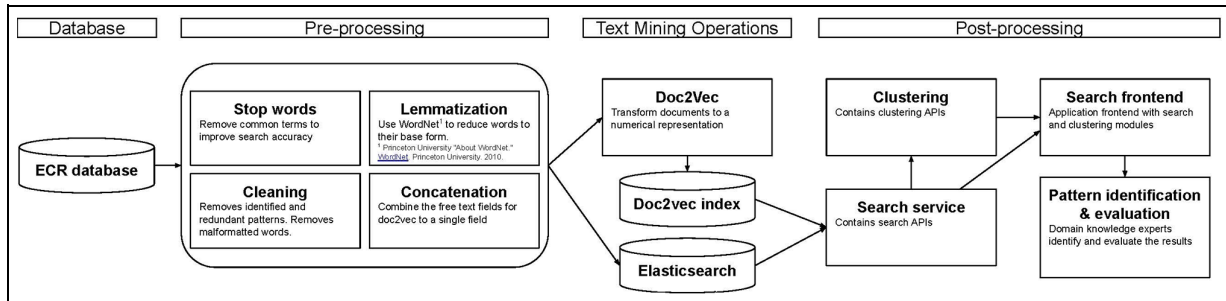
Figure 1. Process flowchart of the methodology used.

and Sojka, 2010) on the normalized data. Third, this model was used by the search service that was given a query, transforming the said query into a document vector that could be matched against the embedded documents of the model.

For the search engine (i.e. standard search methods that rely on matching exact words), the pipeline is somewhat simpler. The search engine used was Elasticsearch (2018) that comes with all the functionalities required to build a search-based application. Similar to the normalization step in the doc2vec model, the input data to the model was lowercased. Additionally, all the stop words were removed from the text to remove noise otherwise caused by these common terms. As part of this project, Elasticsearch provides some benchmark results to validate the doc2vec model. While Elasticsearch serves as the baseline for a typical information retrieval task like the one presented in this paper, the doc2vec model represents a novel approach to information retrieval tasks of this kind. The doc2vec model is simple to use, and because it produces a numerical representation of the text, the output can be combined with any other data mining or feature extraction method that also produces numerical data. The doc2vec model can, therefore, be utilized not only for information retrieval tasks but also for other analytical tasks involving ECRs, such as classifying or clustering.

After the search is performed and a list of documents is returned from either the search engine or the doc2vec model, these documents are sent to a clustering algorithm. In this study, we utilized the LDA for clustering the documents. LDA is a generative statistical model, a form of unsupervised learning that views documents as bags of words (i.e. order does not matter). The method tries to find a number of topics that represent the different topics in the document list.

The results from the algorithm is a set of clusters (i.e. a set of topic words) and each cluster's corresponding documents. The results are displayed in the developed frontend application where the user can easily click around, analyze the document clusters, and determine how they are connected to each other.

The results were then validated together with a domain knowledge expert from the case company, as the text in the documents is a unique case. The domain expert examined the documents for each cluster and assessed whether they are related to the cluster. We focused on evaluating whether the results are relevant up to four documents in each cluster.

Results

Document clustering

In the study, we had 8,000 documents available for matching against queries for NLP and document clustering. To evaluate unsupervised methods (i.e. clustering) we need to know some ground truth describing document clusters. Such did not exist in the database

Table 3. The three words queried horizontally and vertically matching five clusters for each query. At the intersection of each query and cluster, the description labels are given.

Query input	Query output				
	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5
Break	("term," "operation," "ftb," "viewer," "prosit," "light," "park")	("exhaust," "hose," "incident," "oil," "hours," "leakage," "repair")	("acc," "target," "dacu," "lane," "pvt," "fcw," "truck")	("cruise," "control," "engine," "stalk," "icon," "speed," "position")	("air," "popup," "pressure," "park," "door," "leak," "parameter")
Cooling	("leak," "clearance," "bumper," "bolt," "cabinet," "coolant," "air")	("leak," "harness," "coolant," "bracket," "frnt," "closing," "position")	("package," "term," "leak," "coolant," "hardware," "hood," "bolt")	("performance," "temperature," "air," "pipe," "heat," "compressor," "leak")	("pipe," "compressor," "performance," "portion," "heat," "temperature," "package")
Leakage	("cap," "incident," "plug," "repair," "hours," "oil," "engine")	("cap," "plug," "valve," "egr," "pipe," "block," "air")	("durability," "phase," "pto," "code," "configuration," "seal," "shaft")	("repair," "oil," "hours," "cap," "plug," "cylinder," "data")	("valve," "logger," "egr," "hole," "pto," "configuration," "temperature")

we considered here, so the approach was, instead, to test a few queries and let a domain expert validate whether the clusters differ and whether the documents connected to each cluster match the topics describing the cluster.

Three queries were tested using five clusters for each query. Each cluster had seven labels describing the cluster (see Table 3). The number of labels and clusters can be specified at query time, individually for each query. The labels were generated by the document clustering algorithm (i.e. LDA) and should summarize the main key words connected to the clusters.

Each cluster had four reports that were evaluated by a company expert to determine if they belonged to the cluster. All four documents in the three clusters were evaluated as regards their relation to their respective clusters.

For example, the company expert found that documents in Cluster 2 for the search term "break" discussed problems and issues regarding the actual breaks on the truck, while the documents in Cluster 4 discussed the actual breaking maneuver. The expert was, therefore, able to quickly locate four reports in each cluster that were all related to the same topic.

Frontend interface

Another objective of the study is to develop a user-friendly GUI where users can easily make queries and obtain and delve into the results provided. The application is web-based and built on state-of-the-art frontend technologies, such as Node.js (2019) and React (2019).

Figure 2 presents a list of documents related to the search term "break." Users can easily click on a document and get more information directly from the list. The responsive application ensures that users can easily update their queries and directly obtain new search results. In Figure 3, the clustering results for the same query is presented. Here, users can see the clusters that the algorithm found and delve into the documents related to each cluster. Actual data on function group, handler, issuer, and report title have been grayed out in Figures 2 and 3 due to confidentiality.

Discussion

This section considers the research questions in relation to the findings and transferability of results.

RQ1: Can NLP and document clustering algorithms be utilized for grouping ECRs?

Clustering of documents has gained more attention the last years due to better algorithms and larger datasets
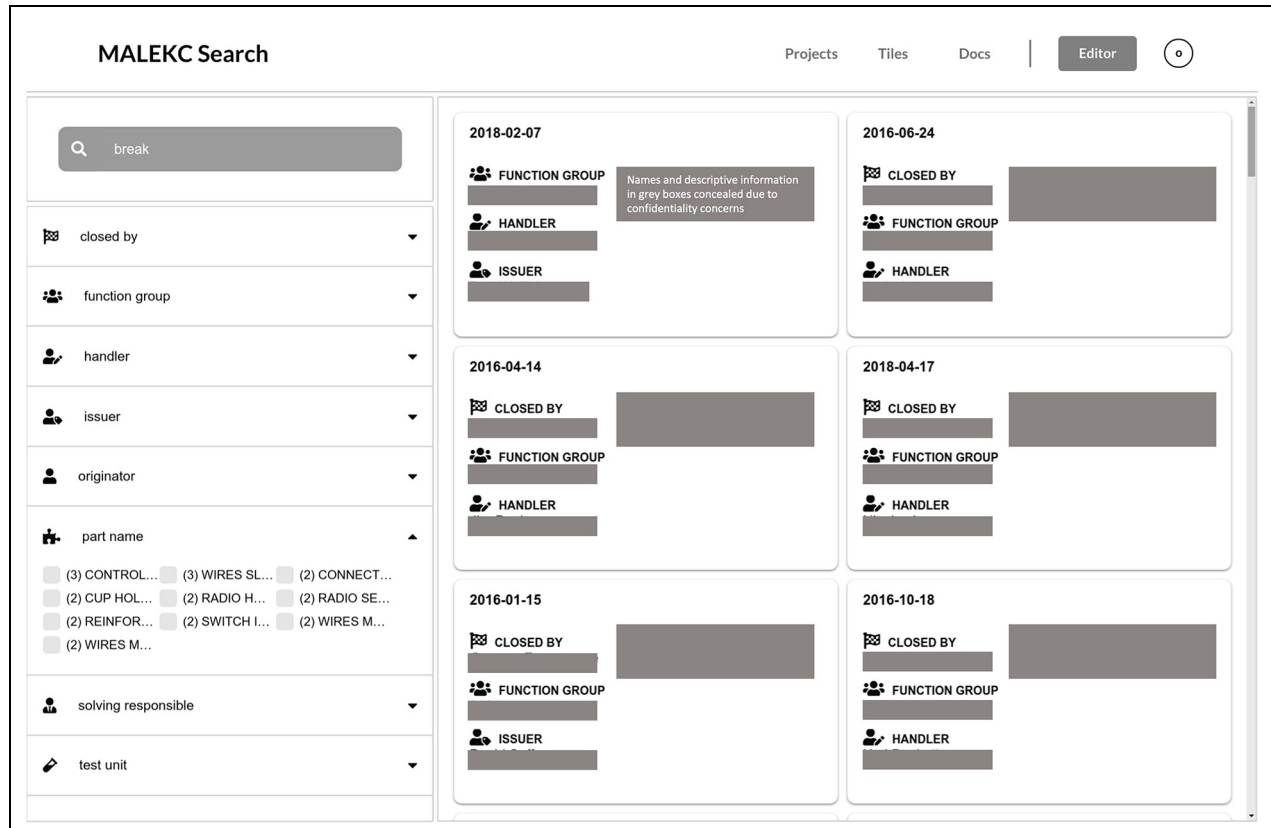
Figure 2. Search application with results list.

available. However, the suitability of these clustering techniques for grouping ECRs that contain very domain-specific language is not directly apparent, and the usual stop words and lemmatization techniques are not optimized.

In this study, we found that document clustering techniques, in our case the LDA, can be applied to ECRs and that the results are useful. The domain expert validated the clusters found via the LDA method and confirmed that they do summarize the main topics that the ECRs discuss.

During the study, other document clustering techniques were also analyzed and tested. For example, standard clustering methods (e.g. DBSCAN, k-means) were utilized for the numerical representations derived from the Doc2Vec model and the latent semantic analysis (LSA). However, the most interesting results were found with the LDA, and this is the reason that the results presented are only the LDA clusters. The results from the other clustering techniques were often very random in behavior and it was not clear why specific documents belonged to the same clusters. To obtain even better results in the LDA clusters, more effort should be spent in cleaning the data from unnecessary information and domain-specific stop words present in all documents.

RQ2: How do product developers benefit from automated document clustering?

Using the NLP and document clustering algorithms tested, clusters of similar documents can be identified automatically. Traditionally, product developers type in a search query in the form of structured or unstructured text. The search results are then displayed in an ordered list by a selected variable that can be filtered by structured text. The main benefit of the approach employed in this study is that when product developers type in a search query, they receive a list of documents that is already clustered into groups. This approach saves time for users since similar documents appear together in respective clusters. There are a few use cases for this approach:

- *Knowledge management*: When making design guidelines with best practice designs, this approach can help to identify related design areas on similar
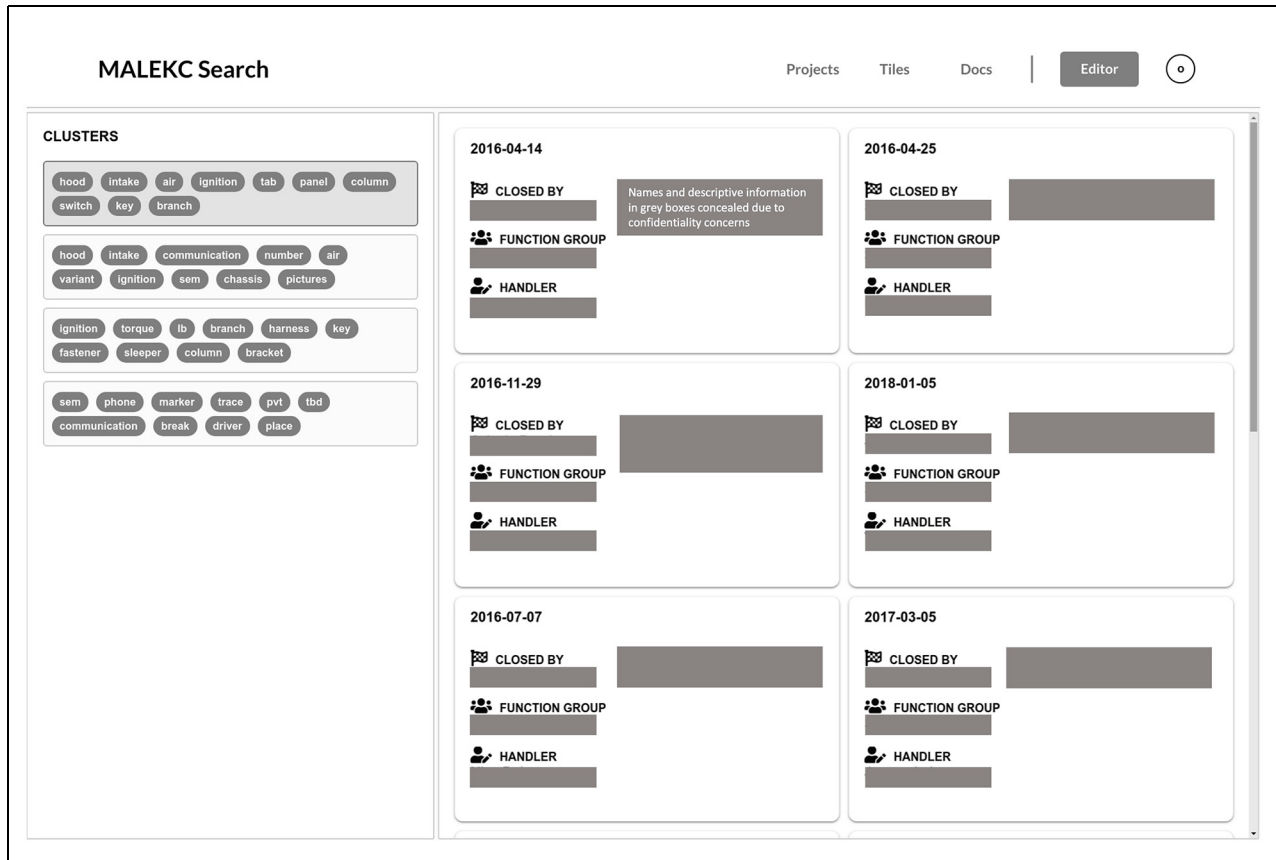
Figure 3. Clustering application.

topics so that they can be further documented for future designs.

- *Inexperienced product developers*: There is often a risk that inexperienced product developers do not have the knowledge of previous work or do not know where to locate these documents. They can use this approach to provide them an insight into similar product features that have previously been designed so that these designs can be taken into consideration when they work on a new product or on redesigning an existing product.
- *Design issues*: When product developers confront design issues, whether these are related to product quality or development, they can leverage a search like this to quickly identify previous designs in the database they can use for root cause analysis. The amount of documented designs makes this task challenging. The approach we proposed can help address this challenge by identifying clusters of related designs for users.

Product developers can take advantage of the document clusters to quickly identify documents of value to a specific situation. From an ECR perspective, this helps to identify historical issues related to a current issue and ensure better pre-studies on historical documents before making a new product or redesigning one. It usually takes product developers a few hours to make a manual list of related documents, but using a model like this reduces that time to minutes.

Limitations and transferability of results

Based on the findings of this study, NLP and document clustering generates some promising results. Given that we managed to find four related documents for each of the five clusters, it is safe to say that we can work further with the results to verify if the model can retrieve more than four documents per cluster or to see if more relevant documents do exist. The reliability of the models can be further improved by performing feedback loops on the model with the cases experts consider to be unrelated. Accordingly, the model can learn that the connections it has made previously are irrelevant.

As mentioned in the research approach, the set-up of this research project in a close partnership with the case company access to real data and domain knowledge experts. The single case study research design, including

the analysis of only one ECR database, enables only limited claims to the transferability of results. However, we show in Arnarsson et al. (2019) that the main steps of approach can be applied to search in a combination of an ECR database and a design guideline database. Nevertheless, additional case studies are needed. These case studies should both examine additional ECR database, and perform more extensive evaluations of each database, that is, with a larger number of queries and variation of clustering criteria.

How, it is difficult to rigorously evaluate any search application since the evaluation criteria depend on the use case. In this study, we performed manual evaluation on each query output from the documents clustering. Conversely, throughout the study, we found no evidence indicating that the method would not be applicable to similar studies that entail data analysis and those that aim to find similarities among text-rich documents. We, therefore, believe that the study can be transferred to most cases where unstructured data is accessible. The field of study should not matter since the algorithms query document text.

Regarding scalability, doc2vec scales linearly with the number of documents, which means that if we double the number of documents in the index, the response time to a query is doubled as well. Elasticsearch focuses on scalability; it has a distributed index that can scale data up to the petabyte scale. In essence, the response time for a query is relative to the number of documents indexed.

Conclusions and future work

Overall, NLP and document clustering approaches seem to work well together in finding and clustering similar documents in an ECR database. The search queries used consisted of one word to define a clear direction as to what information users want to obtain. A demonstrator application was created to demonstrate the method within industry and evaluate the value of such a model to a product developer.

The proposed approach of querying words related to engineering documents and clustering them according to similarities with the use of NLP and document clustering algorithms has potential benefits. The approach was performed on an ECR database, and so far, we found no limitation to including additional databases into the pipeline of the search service.

The study affirms that NLP and document clustering algorithms work in the cases tested, but further tests are needed to explore their limits. The results in Table 3 show the cluster labels assigned by the document clustering algorithm; behind each labeled cluster, there are

ECR reports. The company expert performed a manual evaluation of the results and confirmed that that the four documents related to each cluster are relevant. More cases need to be explored to determine if there are instances when the algorithm starts to work more inefficiently and returns unrelated documents.

Some promising continuations to this work are to test the application with more databases as an input to see if document clustering work across different databases. Also, to expand the number of reports in each cluster and evaluate some potential limits of the approach. This project has focused on only using the free text data included in the documents, which means that future work includes analyzing how it is possible to incorporate also the numerical and categorical data in the documents. By using this information when either training the NLP and Doc2Vec models (i.e. providing context to the documents) or by using the numerical/categorical data when measuring the similarity between documents (e.g. providing that documents concerning the same specific vehicle part should be associated with each other even if their free text data is dissimilar according to the model) could improve the model. Yet a further step forward would be to investigate techniques for identifying similarities amongst shape-based information elements, such as sketches, 3D models, photographs, and videos that are frequently used to document problems and possible solutions in ECRs.

Lastly, in order to achieve better results from the document clustering methods one needs to perform more and better data cleaning. Even though standard NLP cleaning (stop word removal, lemmatization) is performed, one needs to perform more domain specific cleaning of the ECRs in order to really extract the meaning of the free text in the ECRs. Such cleaning might include tasks such as automatic spelling correction and domain specific synonym normalization.

ORCID iDs

Ivar Örn Arnarsson ⓘ https://orcid.org/0000-0002-6935-0143
Johan Malmqvist ⓘ https://orcid.org/0000-0002-4689-4535

References

Aggarwal CC and Cheng XZ (2012) A survey of text clustering algorithms. In: Aggarwal CC and Cheng XZ (eds) *Mining Text Data*. Boston, MA: Springer, pp.77–128.

Ahmad A and Hashmi S (2016) K-harmonic means type clustering algorithm for mixed datasets. *Applied Soft Computing* 48: 39–49.

ArnarssonÍÖ, Frost O, Gustavsson E, et al. (2019) Supporting knowledge re-use with effective searches of related engineering documents: A comparison of search engine and natural language processing-based algorithms. In: *Proceedings of the 22st international conference on engineering design (ICED 19)*, Delft, The Netherlands, 5–8 August 2019.

ArnarssonÍÖ, Gustavsson E, Malmqvist J, et al. (2017) Design analytics is the answer, but what questions would product developers like to have answered? In: *Proceedings of the 21st international conference on engineering design (ICED 17)*, Vancouver, Canada, 21–25 August 2017.

Bird S and Loper E (2004) NLTK: The natural language toolkit. In: *Proceedings of the ACL 2004 on interactive poster and demonstration sessions: Association for computational linguistics*, Barcelona, Spain, 21–26 July 2004.

Blei DM, Ng AY and Jordan MI (2003) Latent Dirichlet allocation. *Journal of Machine Learning Research* 3: 993–1022.

Catron BA and Ray SR (1991) ALPS: A language for process specification. *International Journal of Computer Integrated Manufacturing* 4(2): 105–113.

Dong A (2005) The latent semantic approach to studying design team communication. *Design Studies* 26(5): 445–461.

Dong A and Agogino AM (1997) Text analysis for constructing design representations. *Artificial Intelligence in Engineering* 11(2): 65–76.

Eckert C, Clarkson PJ and Zanker W (2004) Change and customisation in complex engineering domains. *Research in Engineering Design* 15(1): 1–21.

Eckert CM, Stacey MK and Clarkson PJ (2000) Algorithms and inspirations: Creative reuse of design experience. In: *Proceedings Greenwich 2000 international symposium: Digital creativity*, University of Greenwich, London, pp.1–10, 13–15 January 2000.

Elasticsearch BV (2018) The heart of the elastic stack. Available at: https://www.elastic.co/products/elasticsearch (accessed 29 November 2019).

Fayyad U, Piatetsky-Shapiro G and Smyth P (1996) From data mining to knowledge discovery in databases. *AI Magazine* 17(3): 37–54.

Grimes S (2008) Unstructured data and the 80 percent rule. Available at: http://breakthroughanalysis.com/2008/08/01/unstructured-data-and-the-80-percent-rule/ (accessed 29 November 2019).

Jarratt TAW, Eckert CM, Caldwell NH, et al. (2011) Engineering change: An overview and perspective on the literature. *Research in Engineering Design* 22(2): 103–124.

Kim S and Wallace K (2009) An automatic identification of negation in design documents. In: *ICORD 09: Proceedings of the 2nd international conference on research into design*, Bangalore, India, 1–7 September 2009.

Kobayashi VB, Mol ST, Berkers HA, et al. (2018) Text mining in organizational research. *Organizational Research Methods* 21(3): 733–765.

Le Q and Mikolov T (2014) Distributed representations of sentences and documents. In: *Proceedings of the 31st international conference on machine learning*, Beijing, China, 21–26 June 2014.

Mikolov T, Chen K, Corrado G, et al. (2013) Efficient estimation of word representations in vector space. Available at: https://arxiv.org/abs/1301.3781 (accessed 29 November 2019).

Misra H, Cappé O and Yvon F (2008) Using LDA to detect semantically incoherent documents. In: *Proceedings of the 12th conference on computational natural language learning, Association for Computational Linguistics*, Manchester, United Kingdom, 16–17 August 2008, pp.41–48.

Newman ME (2004) Fast algorithm for detecting community structure in networks. *Physical Review* 69(6): 066133.

Node.js. (2019) Available at: https://reactjs.org/ (accessed 22 November 2019).

Pikosz P and Malmqvist J (1998) A comparative study of engineering change management in three Swedish engineering companies. In: *Proceedings of the DETC98 ASME design engineering technical conferences*, Atlanta, Georgia, 13–16 September 1998, pp.78–85.

Prasad B (2016a) Lean, integrated & connected framework for developing smart products. In: Batalla JM, Mastorakis G, Mavromoustakis CX, et al. (eds) *A Handbook on "Beyond the Internet of Things: Everything Interconnected."* Berlin: Springer-Verlag.

Prasad B (2016b) Why are companies embracing on-line web-enabled product development strategies for manufacturing. In: *Proceedings of the 28th international conference on CAD/CAM, robotics and factories of the future*, Kolagat, India, 6–8 January 2016.

React (2019) Available at: https://reactjs.org/ (accessed 22 November 2019).

Rehurek R and Sojka P (2010) Software framework for topic modelling with large corpora. In: *Proceedings of the LREC 2010 workshop on new challenges for NLP frameworks*, Valletta, Malta, 22 May 2010, pp.45–50.

Sarkar S, Dong A, Henderson JA, et al. (2014) Spectral characterization of hierarchical modularity in product architectures. *Journal of Mechanical Design* 136(1): 1–12.

Steinbach M, Karypis G and Kumar V (2000) A comparison of document clustering techniques. *KDD Workshop on Text Mining* 400: 1–2.

Tripathy A and Eppinger SD (2013) Structuring work distribution for product development organizations. *Production & Operations Management* 22(6): 557–1575.

Upshall M (2014) Text mining: Using search to provide solutions. *Business Information Review* 31(2): 91–99.

Wu X, Zhu X, Wu GQ, et al. (2014) Data mining with big data. *IEEE Transactions on Knowledge and Data Engineering* 26(1): 97–107.

Yang N, Yang Q and Yao T (2018) Clustering organization structure in product development projects using similarity. In: *International dependency and structure modeling (DSM) conference*, Trieste, Italy, 15–17 October 2018, p.59.

Yang Q, Lu T, Yao T, et al. (2014) The impact of uncertainty and ambiguity related to iteration and overlapping on schedule of product development projects. *International Journal of Project Management* 32(5): 827–837.

Yin RK (2013) *Case Study Research: Design and Methods*. London: SAGE Publications.

Yoon J, Seo W, Coh BY, et al. (2017) Identifying product opportunities using collaborative filtering-based patent analysis. *Computers & Industrial Engineering* 107: 376–387.

Zhang Y, Chen M and Liu L (2015) A review on text mining. In: *2015 6th IEEE international conference on software engineering and service science*, pp.681–685. New York, NY: IEEE, September 2015.

Author biographies

Ivar Örn Arnarsson is an industrial PhD student *at* the Department of Industrial and Material Science, Chalmers University of Technology in Gothenburg, Sweden. He received his master's degree from the same university in Quality and Operations Management in 2015. He performs his research in close collaboration with a company in the automotive industry. The main research topics for his thesis are Product Development, Quality Management and Data Analytics. Ivar also supervises master students in product development.

Otto Frost is a research engineer at the Systems and Data Analysis group at Fraunhofer-Chalmers Centre, Gothenburg, Sweden. He received his master's degree from Chalmers University of Technology in Computer Science and is currently performing research in Natural Language Processing and Big Data analytics.

Emil Gustavsson is business area leader for Machine Learning and AI at Fraunhofer-Chalmers Centre, Gothenburg, Sweden. He has a PhD in mathematical optimization from Chalmers University of Technology and is currently performing research in Machine Learning, AI, and optimization.

Mats Jirstrand is head of department Systems and Data Analysis at Fraunhofer-Chalmers Centre, Gothenburg, Sweden. He received his MSc degree in applied physics and electrical engineering 1994 and PhD degree in automatic control 1998, both from Linköping University. In 2004 he was appointed associate Professor in automatic control at Chalmers University of Technology. He is the author of over 70 peer-reviewed publications on modelling, simulation, and control system design both for technical and biological applications. His research interests include dynamical systems modelling and analysis, statistical machine learning, and big data analytics.

Johan Malmqvist is a chair Professor in Product Development at Chalmers University of Technology. His research addresses development methodologies and IT support for product development (PLM). Current research focuses on methods and tools for development of product-service systems, for product configuration and for strategic development of PLM solutions. Another area of interest is engineering education. Malmqvist was one of the co-founders and active in the international Conceive-Design-Implement-Operate (CDIO) Initiative, the engineering education model that has been developed by the CDIO Initiative has been adapted by a large number of across the world.