DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Data Engineering and Analytics

Early Detection of Issues in New Car Models using Natural Language Processing on Customer Call Data

Dilara Ademoğlu

DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Data Engineering and Analytics

Early Detection of Issues in New Car Models using Natural Language Processing on Customer Call Data

Frühzeitige Erkennung von Problemen bei neuen Automodellen mithilfe von natürlicher Sprachverarbeitung bei Kundenanrufdaten

Author:	Dilara Ademoğlu
Supervisor:	Prof. Dr. Florian Matthes
Advisor:	Anum Afzal
Submission Date:	17.10.2022

I confirm that this master's thesis in data engineering and analytics is my own work and I have documented all sources and material used.

Munich, 17.10.2022 Dilara Ademoğlu

Non-disclosure notice

The Academic Work contains confidential information that is subject to secrecy. Therefore, the Academic Work must not be duplicated or published without the prior written approval of BMW AG. This requirement of approval terminates on: 10.01.2027

Sperrvermerk

Die Arbeit beinhaltet vertrauliche Informationen, die der Geheimhaltung unterliegen. Aus diesem Grund darf die Arbeit ohne vorherige schriftliche Zustimmung der BMW AG nicht vervielfältigt oder veröffentlicht werden. Dieser Zustimmungsvorbehalt endet am: 10.01.2027

Acknowledgements

I want to thank Prof. Dr. Florian Matthes for supervising this thesis and giving me the opportunity to work with his chair. I am also very grateful to my advisor M.Sc. Anum Afzal for her constant advice throughout the project, and also M.Sc. Tim Schopf for his insights into my work. I appreciate their availability for discussion, suggestions, and guidance.

I express my gratitude to Marcel Eigner and Stephanie Seisenberger Glenn from the BMW Group for being incredible team leads. Without their support, this research work would not have been possible. I also would like to thank my company supervisor Ferdinand Kleber for his availability and support in understanding the technical data and use cases at BMW Group and for guiding me through the business processes, as well as Andreas Matula from BMW Group for his help and assistance.

I would like to thank my family for their encouragement during difficult times, and all my friends, especially Aynur, Pınar, Theo, Andrea, İrem, Yasin, Piotr, and Fabien, for their constant support during my work on this thesis and for being there for me whenever I needed them.

Abstract

Early detection of quality issues has been an important topic that can give a significant competitive advantage to businesses. Many automotive companies, including BMW Group, conduct tests during production to detect potential production errors. Even though a vehicle passes all the tests, some issues can go unnoticed and occur on the customer's side after the purchase. BMW Group seeks to detect possibly unnoticed issues to improve production quality by inspecting customer call data. The aim is to improve the conducted tests along with the vehicle quality and prioritize the known issues on their frequency and importance, with the end goal of improving overall customer satisfaction. Customer call data is collected regularly through call centers to get feedback on customer satisfaction; this type of data is considered unstructured data. The annotation of this type of data is expensive and time-consuming, making it challenging to apply supervised machine learning methods.

This thesis aims to implement Natural Language Processing (NLP) techniques as part of the root cause analysis. The NLP techniques used for the project are topic modeling and context-based text matching. We compare two topic modeling approaches, one traditional and one embedding-based approach, to get topic representations of customer call data and production data. Furthermore, we discuss which representation is preferable. Selected topic representations are later compared using similarity measures to match similar topics between the two datasets. An analysis is conducted on the findings of the topic matching task to discover the recurring issues in customer call data that are not stated under the production error data. We evaluate our findings with topic coherence measures and with a user study. With the help of state-of-the-art NLP models and data analysis, we demonstrate that potential issues reported by customers can be detected and mapped to the corresponding issues in the production error data, which is crucial for prioritizing issues and customer satisfaction.

Contents

1. Introduction

Customer call data has been a vital feedback source for many vehicle production companies, including BMW Group. Analyzing customer feedback provides insights into the customers' wants and needs, along with helping grow and maintain the customer profile. Within BMW Group, customer feedback is used to monitor satisfaction among customers, as well as to detect reported defects to improve the production of new car models. However, due to the size of the data and the rapid growth of the number of data points, it has been difficult to analyze the data by hand.

In this work, we use Natural Language Processing methods such as topic modeling and text similarity to determine the occurring issues in vehicle production through customer call data. Even though we have encountered other studies on early detection of issues in the automotive industry by applying regression models to vehicle datasets[1], research on issue detection is conducted using neither NLP methods nor customer feedback datasets for the automotive industry. We aim to find the undiscovered issues by matching the already occurring issues in the customer feedback data with the production error data.

1.1. Motivation

The quality control departments at BMW Group want to gain as many production-related insights as possible from the customer feedback datasets collected through customer calls. They aim to compare this with the already existing production error datasets in order to prioritize the known issues better and enhance the conduction of quality tests, eventually leading to increased customer satisfaction. However, due to the time-consuming nature of the analysis of unstructured datasets and the difficulty of keeping track of the fast growth of these datasets, it is rather challenging and unscalable to manage this task manually.

Even though many tests and analyses occur during production, this process can be prone to error due to human nature. Deciding on which of the known errors are needed to be prioritized is an important aspect of improving the production systems and finding out the possibly unnoticed errors. We aim to improve the production systems using the information extracted from customer feedback, which is a valuable source of information. BMW Group values the opinions of its customers and aims to use this information to enhance customer satisfaction. The customer feedback dataset provided by BMW Group is not labeled; hence it requires unsupervised NLP methods to be analyzed. In order to extract as helpful information as possible, topic modeling and text similarity approaches are proposed.

1.2. Processes

Before presenting the problem statement, it is necessary to present two processes from BMW Group. Under the 'Quality data', which is the data asset that covers every data process relevant to the quality of a vehicle, there are several datasets and processes, two of which are used for this study: Welcome Call Process and Problem Quality Management (PQM) Process.

1.2.1. Welcome Call Process

After a vehicle is purchased, it is conventional to get feedback from customers at BMW BMW Group. The customer gets a call from a customer satisfaction employee 45-50 days after purchasing the vehicle. They get asked about their initial experience with their new vehicle. The conversation is then recorded as a transcript. This process is called the Welcome Call process, shown in Figure 1.1, which is conducted in the US market, so the collected data is in the English language. The dataset is later used to analyze the overall satisfaction of customers, which is elaborated more on chapter 3.

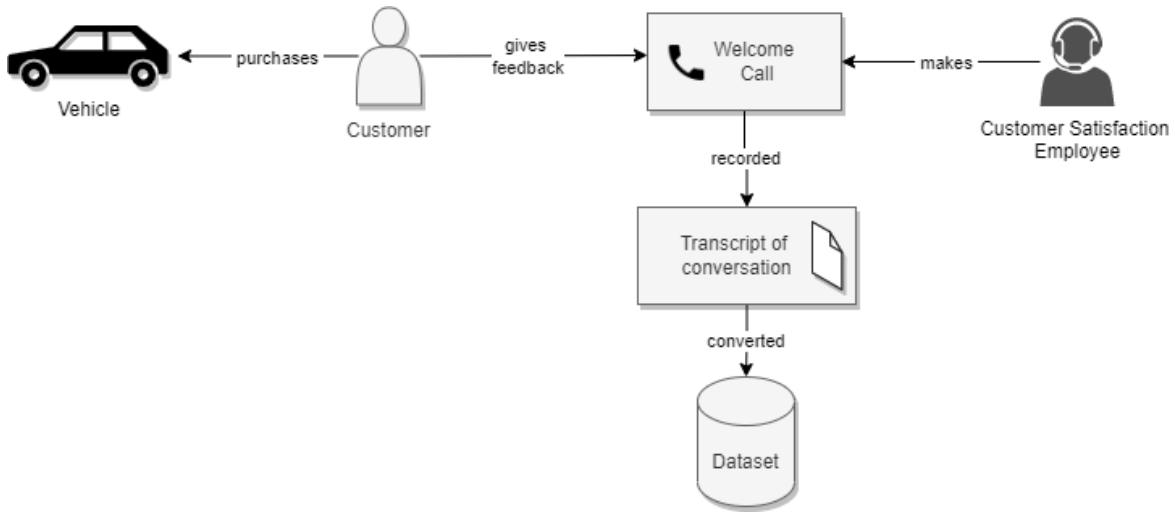


Figure 1.1.: Welcome Call Process. The customer gets a call from the customer satisfaction employee if they consent to the conversation being recorded; this is converted to a transcript, which is later stored on a dataset.

1.2.2. Problem Quality Management Process

In BMW Group, customer-relevant digital product topics and hardware issues occurring in the vehicle during production appear in Problem Quality Management (PQM) data, which is the production error data that is mentioned earlier. PQM Process is a data asset that is part of Root Cause Analysis in BMW plants. Two objects that are used for this study are PQM Incident and PQM Problem.

The PQM Incident represents a collection of information created by a sensor as part of the error recording. The PQM Problem is generated during error handling by the plant employees. It includes the relevant PQM Incidents as well as the results of problem processing. A new PQM Incident is assigned to a PQM Problem if it is suitable. A new PQM Problem is created if unassigned PQM Incidents can be grouped together; hence not every PQM Incident is assigned to a PQM Problem. Figure 1.2 shows the process from generation to the dataset. Both datasets are used for many business processes as the source of quality control analysis of production lines.

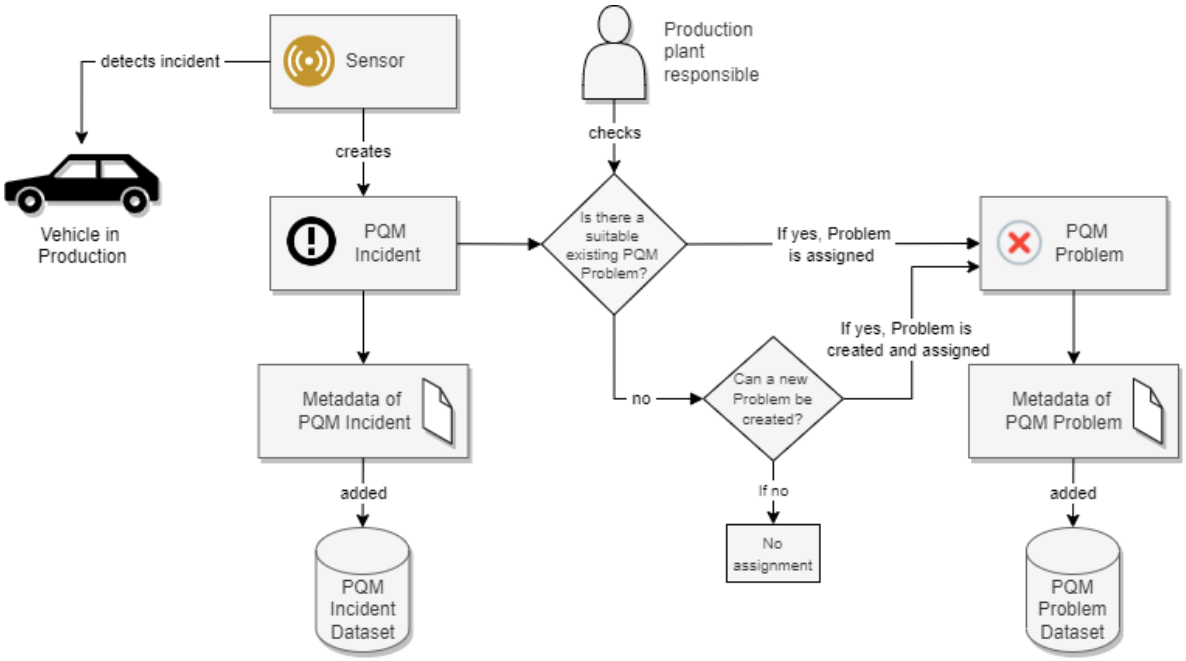


Figure 1.2.: PQM Process. Vehicles go through many tests with sensors. If a sensor detects an incident, it creates an incident report, which is then recorded in the dataset. Incidents do not have an assigned problem by default. If the incidents start to grow, a production plant employee creates a PQM Problem, and the relevant incidents are assigned to this problem. If there is an already existing PQM Problem, the incident is assigned to that instead.

1.3. Problem Statement

To extract the occurring issues from the customer feedback dataset, we used topic modeling, which is an unsupervised NLP technique to analyze text by clustering groups of words to determine the underlying structures of text data. Through the extracted topics, it is possible to examine the occurring issues through feedback provided by customers. However, depending on the choice of topic modeling method, the generated topic words can vary. Some traditional methods like Latent Dirichlet Allocation (LDA)[2] and Latent Semantic Analysis (LSA)[3] can

generate good topic representations; however, with the advancing research on deep learning techniques, NLP methods have also adapted to benefit from more complex approaches such as word embeddings[4]. Taking contextual information of text data has improved many NLP tasks along with the topic modeling outputs [5]. Especially for domain-specific research, such as this study, generating meaningful document representations could be challenging.

Document matching is a recurring NLP task that has many methods proposed as solutions. In this study, we aim to match the topics generated from the documents. Matching the document representations requires applying similarity measures such as the well-known Cosine Similarity [6]. It is a simple formula that compares two matrices in the vector space, which researchers have improved over time by adding the element of word embeddings into the equation [7]. Using the word embeddings as the vectorized text input allowed researchers to integrate the semantic information of the language into the similarity calculations, and with the emergence of contextual embedding methods such as BERT[8], the context of the input sentences was also taken into account.

The solution created by machine learning methods can be complex with many technologies; however, at the end of the day, what matters is how much value it adds to the business [9]. Presenting the results of a scientific study to users of different domain-expertise on business processes is not an easy task. Topic modeling visualizations mainly focus on Word Clouds, which use the extracted topic keywords from the documents. Other ways to visualize topic models include intertopic distance maps of topic clusters, bar charts of topic frequencies, heatmaps of topic similarity, and dendrogram for hierarchical inspections, depending on the used method [10].

1.4. Research Questions

This work focuses on the following research questions:

1. Which state-of-the-art topic modeling approaches would provide better insight into BMW customer feedback datasets?
2. Which text similarity techniques give better matches between BMW customer feedback datasets?
3. How to support quality control departments with interactive topic visualizations of BMW customer feedback datasets?

1.5. Proposed Solution

BMW Group wanted to digitalize the analysis of customer feedback datasets to help improve the production systems and understand the underlying structures in customer call data. For this purpose, it was proposed to design a tool that uses NLP techniques to work with unstructured text data. The designed tool is a web application that is built with Flask [11]. The back-end of the tool contains an NLP-Engine with a preprocessing pipeline for

the datasets, two topic models, one for each dataset, that is chosen based on the results of experiments explained in chapter 5. The tool contains visualizations of the explored topic representations, which are later evaluated by a user questionnaire.

For NLP techniques, topic modeling was proposed to find the underlying issues through the customer feedback data, and text similarity was used to match the issues found in customer feedback data with the production error data. An analysis was conducted to filter out the issues that were not observed in the production error data. For topic modeling, we compared two methods, Latent Dirichlet Allocation (LDA)[2] as the traditional approach and BERTopic[10] as an embedding-based approach. We applied the methods to both datasets, with different parameter settings, and explored which approach suits this problem better. In order to get better matches between the topics, we explored variations of Cosine Similarity with word embeddings and contextual embeddings. We present our findings on the newly discovered issues that are not covered by the production error datasets and elaborate on the quality of matches as the conclusion of this study.

The thesis is organized as follows: In chapter 2 theoretical background is presented. Datasets used in this study are explained in chapter 3, as well as the preprocessing methods. Methodology is explained in chapter 4, experiments and results are reported in chapter 5. We explain the system architecture of the created web application on chapter 6 and present the evaluation results of the application on chapter 7. Limitations and future work has been discussed in chapter 8 and we conclude our work on chapter 9.

2. Background

In this chapter, we describe the basic theoretical concepts required for understanding the work done in this study. We start with Natural Language Processing topics, then move forward to clustering methods. We explain the necessary background on Topic modeling and Text Similarity and finish the chapter with Customer Feedback Research.

2.1. Natural Language Processing

Natural language refers to how people communicate with each other. The main elements in natural language are speech and text. Since computers are unable to understand texts as humans do, computerization of text data requires transforming text data into mathematical calculations [12]. NLP is a subfield of Artificial Intelligence that allows computers to understand natural language. Earlier methods are based on statistics and rule-based approaches; however, with the advancements in machine-learning algorithms and deep-learning architectures, state-of-the-art methods for various NLP tasks are becoming more deep-learning based.

2.1.1. Unstructured Data

Datasets can be classified into two groups, structured and unstructured[13]. Structured data has a predefined format to a set structure before being placed in data storage. It is easier to understand structured data for business users and run machine learning algorithms on the data. Unstructured data is stored in its native format and not processed until it is used[13]. Unstructured data is accumulated much faster than structured data since there is no need to predefine a structure. Even though the native format of unstructured data provides freedom for a larger pool of use cases, it requires data science methods to extract information from unstructured data. The text has a specific structure, such as grammar, punctuation, and spelling; however, this structure is too complex for computer understanding[14]. That is why text data is considered to be in an unstructured format; hence this study works with unstructured data.

2.1.2. Preprocessing

Text preprocessing is an essential step in the analysis of text data. Some steps for text preprocessing are text cleaning, tokenization, stemming, and lemmatization.

Text Cleaning

Text cleaning is an essential step for many NLP tasks, which is cleaning the words and characters that are considered 'noise' from the text. Some essential steps of text cleaning include expanding contractions and removing stop words, digits, and non-alphabetical characters.

Expanding the Contractions

As part of text cleaning, it is conventional to remove contractions that occur in the text. A contraction is when two words are shortened into one-word form by dropping letters and replacing them with apostrophes. Some examples of expanding the contractions are:

- She'd like to know. ⇒ She would like to know.
- He's going to buy. ⇒ He is going to buy.
- Customer didn't like the feature. ⇒ Customer did not like the feature.

Stop-word Removal

Stop words refer to the common words that occur in a language that does not add information to the sentences. Some examples of stop words would be: '*the*', '*is*', '*this*', '*such*', '*and*' etc.

Lemmatization

Lemmatization is the task of determining that two words have the same root, despite their surface differences [15]. A lemma is the canonical base form of the word. Unlike the stem, which is the part of the word that never changes morphologically and is usually derived by removing the ending of the words, a lemma is derived through the use of vocabulary and morphological analysis of words by removing only the inflectional endings [16]. Given the word '*saw*', a stemmer would remove the suffix and return the string '*s*' as the output, whereas a lemmatizer would return the base form '*see*' for the same word.

Below some examples of lemmatization are given:

- Noun lemmatization
  car, cars, car's, cars' ⇒ car
  mouse, mice ⇒ mouse
- Verb lemmatization
  be, am, are, is ⇒ be
  sing, sings, sang, sung, singing ⇒ sing
  break, breaks, broke, broken, breaking ⇒ break

- Adjective lemmatization
  fruity, fruitful ⇒ fruit
  believable, believing ⇒ believe
  live, living, livable, lively ⇒ live

2.1.3. Text Vectorization

For text data to be machine-readable, it has to be represented in numeric form; that is where vectors play a crucial role. Text vectorization can be achieved through traditional count-based methods or more complex methods such as neural word embeddings. Bag-of-words (BoW) and Term Frequency - Inverse Document Frequency (TF-IDF) can be given as examples of the count-based methods, whereas Word2Vec and GloVe [17] can be named for examples of word embedding-based methods. Count-based methods often result in long and sparse vectors, while embedding-based methods result in shorter and dense vectors.

Bag-of-Words

BoW is one of the traditional text vectorization methods. As the method's name suggests, this representation treats the document as if it were a bag of words. It is an unordered set of all the words that occur in the document, and values in the vector are the frequencies of each word.[15]

Term Frequency - Inverse Document Frequency (TF-IDF)

Although the raw word frequency gathered by BoW representation can be a useful aspect for text vectorization, it is not the best measure of association between words since it is not very discriminative. In a document, some words can occur more frequently than others; therefore, BoW representation would give more importance to these non-informative words, such as stop words. Stop word removal helps with bag-of-words representation; however, TF-IDF has an internal solution to this downside as it attempts to give more chance to the lower frequency words [15]. It can be also considered as a weighted BoW representation of words since the raw count of term frequency ($tf_{t,d} = count(t,d)$) is still used as the first factor, which is squashed into logarithmic form instead of raw form in this method: $tf_{t,d} = log_{10}(count(t,d) + 1)$

The second factor in TF-IDF is called the inverse document frequency. Given the number of documents N and df_t, the number of documents a term t occurs in, the inverse document frequency is calculated by $idf_t = log_{10}(\frac{N}{df_t})$; this measures how much information a term provides based on its frequency over all documents. If a term frequently occurs in all documents, it does not provide significant information to the task; hence the IDF results in zero for that term. The fewer documents in which a term occurs, the higher the weight for that term.

TF-IDF is obtained by taking the product of term frequency and inverse document frequency; with this product, the term frequencies are weighted based on their occurrence

in the documents. The value in the matrix representation for term t would then be $tfidf_{t,d} = tf_{t,d} \cdot idf_t$. The TF-IDF representation will result in documents with similar words having similar vectors, which is useful for machine-learning algorithms.

2.1.4. Word Embeddings

Fixed-length vectors for representing words are called embeddings because the word is embedded in a particular vector space. The classical text vectorization methods such as BoW or TF-IDF mentioned above assume each term in the vocabulary is independent of each other; hence they fail to represent the semantic relationships between terms. Word embeddings can capture semantic and syntactic relationships of the terms occurring in the given text and provide a dense vector representation. The values of these dense vectors will be real-valued numbers that can be negative instead of mostly zero sparse vectors generated by count-based methods. Word embeddings are static, meaning the methods learn one fixed embedding for each word in the embedding vocabulary. Most commonly used word embeddings are Word2Vec [18][19], GloVe [17] and FastText[20]. Among these, Word2Vec and FastText are prediction-based methods that use a neural network architecture, while GloVe is a count-based method that uses matrix factorization. Figure 2.1 shows an example, the distance between 'man' and 'woman' is equal to the distance between 'king' and 'queen'; hence the semantical relation on gender relation is protected.

Word2Vec

Word2Vec is a feed-forward neural network-based model to find word embeddings; it is the first of the neural word embedding models. Two model architectures are proposed for computing continuous vector representations, the Continuous Bag-of-Words (CBOW) Model and the Continuous Skip-Gram Model [18][19].

1. **CBOW Model**
   CBOW model is a simple architecture with one hidden layer. It takes the input surrounding context words and sends them to this hidden layer to predict the current word. For example, given a sentence '*Apple pie is a dessert.*', the model iterates through the words in the sentence. It starts with the word '*apple*' and tries to predict this word by reading the other words in the sentence '*pie*', '*is*', '*a*', and '*dessert*'. Then, it tries to predict the word '*pie*' by reading the surrounding words '*apple*', '*is*', etc. After this training step, to obtain the embedding vector of a particular word, it is fed to the neural network as input, and the hidden layer value is taken as the embedding vector.

2. **Skip-Gram Model**
   Skip-Gram architecture is similar to CBOW, but instead of predicting the current word given the context words, it tries to predict the surrounding context words given the current word. Each current word is used as an input to a log-linear classifier with a continuous projection layer, and the words within a certain range before and after the current word is tried to be predicted. The embedding vector is again obtained by the

hidden layer value after the training by feeding the selected word to the network. The best performing variant of the Word2Vec model was introduced in [19] as a Skip-Gram model.

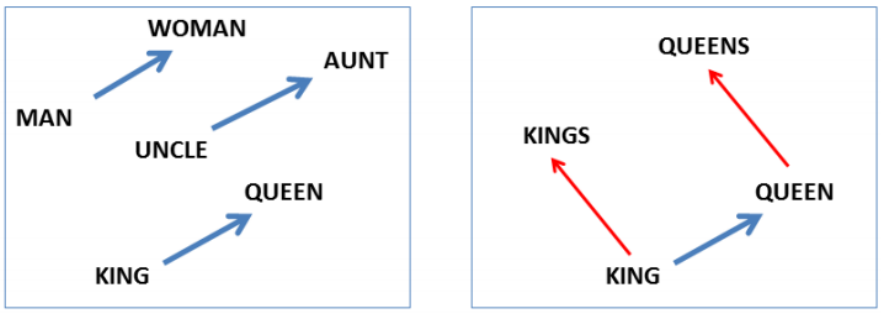


Figure 2.1.: The left figure shows distances of three pairs of words on gender relations in vector space. The right figure shows plural and singular relations for a word pair. Image taken from [19].

Both of the proposed architectures are simple; instead of a multi-layer neural network, Word2Vec presents a binary classification problem by using a one-layer logistic regression classifier by asking the question 'Is word w_1 likely to show up near w_2?'. However, it does not have a reliable way to deal with out-of-vocabulary (OOV) words that are unseen in the training corpus [15].

GloVe (Global Vectors)

GloVe is an unsupervised learning algorithm for obtaining vector representations for words that uses a log-linear model. Aggregated global word co-occurrence statistics from a corpus are used for the training [17]. In the Word2Vec model, the word co-occurrences carry no additional information; they just create more training samples. However, in the GloVe model, the co-occurrences of context words are important and contain meaning. GloVe model essentially performs similarly to the Word2Vec model and tackles the same problem with a different approach. Rather than raw counts of co-occurrences, it uses ratios of co-occurrences to encode semantic information about word pairs[21]. It combines the advantages of global matrix factorization methods such as LSA and local context window methods such as the Skip-Gram model [19]. The researchers aim to create a model that does not suffer from the drawbacks of these two methods. LSA uses statistical information quite well at the expense of performing poorly on deriving word analogies, whereas the Skip-Gram model performs well in the word analogy task but fails to understand the statistics of the corpus since it uses a local context window instead of a global one. GloVe model reports achieving better results with the same parameters in a shorter time than Word2Vec [17].

FastText

FastText differs from previously mentioned embedding models Word2Vec and GloVe, as it is a character-level, not a word-level embedding model. It is based on the Skip-Gram model[18] and represents each word as an n-gram of characters. Instead of representing words as vectors, a vector representation is associated with n-grams, and the words are represented as the sum of these n-gram vector representations [20]. Morphologically rich languages that use compositional word-building benefit from the character-level embedding models since character-level representations are helpful to deal with OOV words with their lower-level language elements. They are also preferred due to their small model sizes and fast computation times [21][22]. FastText has been shown to achieve significantly better performance than the Word2Vec tool, as stated by researchers. Besides learning the vector representations of words, it is proven useful for tasks such as text classification and language identification [20][23][24].

2.1.5. Contextual Embeddings

One downside of the previously explained word embeddings is that they only have one single global representation for each word. The context of the words is not taken into account in these representations [25]. Researchers have come up with scientifically revolutionary contextual embedding models such as ELMO[26], BERT[8], and GPT[27] models, that are able to achieve state-of-the-art results for many downstream NLP tasks [28].

Traditional word embeddings try to learn a global embedding matrix with a fixed vocabulary size and specified length for the vector representation. Contextual embeddings capture the uses of words across varied contexts and assign words with token-level representations based on their context. Most contextual embeddings are based on Transformer architecture[29] that uses a self-attention mechanism with an encoder-decoder architecture. Transformer models have shown great success for many text generation tasks such as machine translation, text summarization, and question answering [25].

BERT

Bidirectional Encoder Representations from Transformers (BERT)[8] is a pretrained language model that proposes a Masked Language Model (MLM) pretraining objective as an improvement to its predecessor language models that uses fine-tuning approach such as GPT [27] model, which uses a left-to-right architecture, that restricts each token to only be able to attend to previous tokens in the self-attention layers. The proposed masked language model randomly masks some of the tokens from the input sequence and aims to predict the masked tokens, which allows the pretraining of a bidirectional Transformer[29] by enabling the representation to concatenate left and right contexts. Aside from MLM, BERT uses Next Sentence Prediction (NSP) as the second pretraining objective to improve the performance of downstream tasks. Given two input sentences, NSP predicts whether the second sentence is the actual next sentence of the first sentence [25]. A distinctive feature of BERT is that there is minimal difference between the final downstream architecture and the pretrained

architecture[8], which makes it easy to fine-tune the same model for different downstream tasks.

BERT has inspired the research society, and many new variants have been developed based on the BERT architecture. Among these RoBERTa[30], ALBERT[31], and Sentence-BERT[32] are worth mentioning. With a few changes to the BERT model, RoBERTa achieves considerable improvements. ALBERT argues that the NSP objective lacks difficulty and proposes a sentence-order prediction objective [25]. Sentence-BERT proposes more modifications to the BERT model to provide a solution to finding the most similar sentences. It manages to encode sentence-level semantic meaning and reduces computation time for this problem from 65 hours with BERT to about 5 seconds [1].

2.2. Clustering

Clustering is an unsupervised learning technique that is especially preferred when the dataset is not labeled. This method groups the data points that are similar between themselves, corresponding to the hidden patterns among the data points. Clustering techniques can be grouped into partitioning, density-based, grid-based, model-based, and hierarchical methods, also fuzzy clustering [33]. Partitioning methods, such as K-means[34] and K-medoids, define the clusters by a centroid and classify each data point as a unique cluster. Selection of the cluster number is essential for these methods. Despite the popularity of K-means, it has shown to be sensitive to the shape of the clusters and has a higher possibility of performing poorly. Density-based clustering methods aim to discover clusters of arbitrary shapes that help handle the outliers and noise in data since they do not assign every point to a cluster. Density refers to a number of objects in a neighborhood of data objects. In fuzzy clustering, data points can belong to more than one cluster due to the soft bordering. Decision trees and neural networks can be given as examples of model-based clustering methods. Hierarchical clustering builds clusters gradually by creating a nested cluster structure that can be visualized as a dendrogram. Hierarchical procedures can be agglomerative, which starts with each element as a single cluster and joins them as the algorithm progresses, or divisive, which starts with one single group and splits them into minor clusters [35].

In this study, one of the topic modeling approaches uses a hierarchical clustering approach HDBSCAN[36] as a document clustering step of their architecture. HDBSCAN is an extension of the density-based method DBSCAN[37], which combines hierarchical and density-based methods. Density-based methods are efficient and robust to noise in the data, although they suffer from parameter selection and handling of variable density clusters due to their global threshold choice, which are two problems HDBSCAN improves. HDBSCAN builds up a hierarchy to figure out which peaks end up merging, including the order information, then for each cluster decides on splitting to subclusters or not [2]. Comparative studies between K-means, DBSCAN, and HDBSCAN show that HDBSCAN can tackle the issues with density-based and partitioning methods mentioned earlier [38].

[1] https://www.searchcandy.uk/nlp/sentence-bert/

[2] https://pberba.github.io/stats/2020/07/08/intro-hdbscan/

2.3. Topic Modeling

Topic modeling is a text-mining method that aims to discover hidden semantic patterns given a set of documents. It is an unsupervised method that clusters the documents into meaningful groups. The topic words generated by topic models are clusters of similar words. Topic modeling has gained popularity within the industry, especially with the increasing amount of collection of unstructured data and the method's easy-to-use strategy. It is often used when a large collection of text cannot be reasonably read and sorted through by person [5]; thus, it helps users to understand large document collections. The method can be used to analyze customer reviews, blog posts, newspaper articles, scientific publications, etc.[39]

The traditional methods of topic modeling include methods such as LDA[2], LSA[3], Non-negative Matrix Factorization (NMF), and Correlated Topic Model (CTM)[40]. However, with the advancing research on the deep-learning field and word embeddings, neural topic models such as Top2Vec[5] and BERTopic[10] have been developed and gained popularity [41].

2.3.1. Traditional Approaches

Traditional approaches focus on reducing the dimensionality of documents by using text vectorization methods, mainly BoW and also TF-IDF, both explained in subsection 2.1.3 in detail. TF-IDF has been used as a reduction technique for modeling text corpora in the early research[42]; however, it has some shortcomings, such as the reduction of the size in document representations not being significant and the generated representation of documents not providing enough information[2]. To overcome these shortcomings, researchers proposed approaches such as LSA[3](also known as Latent Semantic Indexing (LSI)) and further improved this method by adding probability when pLSA[43] was proposed. LSA is an algebraic method based on Singular Value Decomposition (SVD) [44], which uses vector representation of text to compute similarity between texts to find similar words. NMF was proposed to deal with the negative components occurring in the data models that do not represent real-world data. pLSA models each word in a document as a sample from a mixture model [45].

LDA [2] is a generative probabilistic model used for the collection of discrete data, which is the traditional topic modeling approach used in this study. It is one of the most popular methods used for topic modeling [44][46] and is commonly used as a baseline by many researchers. LDA assumes that documents are probability distributions over topics and topics are probability distributions over words. The model has three necessary hyperparameters, the number of topic clusters, and two Dirichlet priors, alpha and beta. High alpha indicates that each document is likely to contain many topics, which increases the topic diversity, and high beta means each topic contains a mixture of many words [47]. The number of topic clusters hyperparameter is shown to be the most influential hyperparameter on the model quality [48]. However, LDA is observed to have limited performance when it comes to short documents [49].

2.3.2. Embedding-based Approaches

As the advancement of neural networks progressed, they also started being used for topic modeling tasks, which led to Neural Topic Models. Autoencoders [50][51] and Recurrent Neural Networks [52] are some of the early deep learning architectures that also have been applied to topic modeling problems. After Word2Vec came out, embedding-based topic modeling approaches started to develop. The early methods convert the documents into embeddings and then adapt LDA to generate topic words and clusters. This approach improves the performance compared to the ones using BoW; however, they are still generating topics that are context-independent [53]. With the development of Transformer models, it became inevitable for pre-trained contextual language models such as BERT to be used for topic modeling tasks [54][55][56][57]. Besides using pretrained language models, researchers tried applying clustering methods to the generated contextual embeddings to get topic representations of the documents. One study [58] applied K-means clustering to the outputs of BERT, RoBERTa, and GPT-2 embeddings; they observed that running a simple clustering algorithm on token-level embeddings generated by the pretrained contextual embeddings resulted in word clusters that are similar to the ones outputted by an LDA model. Another study [59] proposes combining known word embedding methods such as Word2Vec, GloVe, and FastText, or contextual embeddings such as ELMO and BERT with clustering algorithms such as K-means, K-medoids and Gaussian Mixture Models. Other studies [10][5] added dimensionality reduction as a middle step before applying clustering to the embedding models. Top2Vec [5] uses doc2Vec[60] as document embeddings, which is an extension of Word2Vec that adds paragraph vectors to the learning task. They create jointly embedded document word vectors, then apply UMAP [61] for dimensionality reduction before finding dense clusters with HDBSCAN. TopClus [53] jointly models topic-word and document-topic distributions with an attention-based document embedding learning model. They argue that the naive approach of dimensionality reduction of document embeddings may not lead to naturally suited input for clustering; hence they jointly learn a spherical latent space projection that is suited for the clustering task with the EM algorithm.

In this study, we used the embedding-based method BERTopic [10], which is similar to the Top2Vec method. BERTopic embeds documents to create vector representations that allow documents to be compared semantically, assuming documents with the same topic are semantically similar. As the pre-trained language model, Sentence-BERT [32] is used to convert sentences and paragraphs to dense vectors. These embeddings are not directly used to generate topics but rather to cluster similar documents together. The generated document embeddings are in high-dimensional space; hence the vector representations are sparse. UMAP is used to reduce the dimension of the document embeddings. UMAP is preferred to other dimensionality reduction techniques such as t-SNE and PCA due to having no computational restriction on reduced embedding dimensions and being able to be used across language models with different dimension requirements. HDBSCAN, explained in section 2.2, is used for clustering the embeddings with lower dimensions from the output of UMAP. Using UMAP before HDBSCAN has improved the performance on both clustering accuracy and time aspects. By using HDBSCAN as the clustering approach, BERTopic prevents

irrelevant documents from being assigned to the nearest cluster, successfully integrates a topic reduction feature, and aims to achieve better topic representations. For extracting topic representations, BERTopic uses a class-based variation of TF-IDF. Instead of using all documents as input to TF-IDF, it treats all documents in a cluster as a single document by concatenating them, denoted as class c. The TF-IDF formula is modified by replacing inverse document frequency with inverse class frequency, formulated as $icf_t = log_{10}(1 + \frac{A}{tf_t})$, where A denotes the average number of words per class. The formula for class-based TF-IDF then becomes $tfidf_{t,c} = tf_{t,c} \cdot icf_t$, where $tf_{t,c}$ refers to the term frequency in the class. By applying this step, BERTopic aims to model the importance of words in clusters rather than in documents, which is then used for generating topic-word distributions for each cluster of documents. One downside of the BERTopic model is that one topic is assigned strictly to one topic, which limits the topic classification for real-world documents.

2.4. Text Similarity

Text similarity measures play an important role in NLP research, such as document clustering, topic modeling, text summarization, etc. Units of text data such as words, paragraphs, and documents can be semantically and lexically similar. Lexical text similarity methods are based on the grammatical structure of the sentences, focusing on word counts, which is useful for word-level similarity; however, it completely neglects the meaning aspect of the text. Over the past decades, semantic similarity measures have been proposed by researchers to overcome the drawback of lexical similarity methods [6]. Knowledge-based semantic similarity methods involve using Lexical databases such as WordNet[62]. However, they rely strictly on the existence of such knowledge databases that need to be updated frequently to fit the current language trends. Corpus-based methods measure the semantic similarity between terms occurring in large corpora. The information can be retrieved from the corpora by using methods such as LSI or LDA [6] explained in section 2.3, or it can be converted into the vector form through methods mentioned in subsection 2.1.3, clustering algorithms such as K-means and K-medoids can be applied to generate clusters [63]. After conversion to a suitable form, various distance measures such as Jaccard Coefficient, Manhattan Distance, Jensen-Shannon Divergence, Kullback-Leibler Divergence, or Cosine Similarity can be applied to measure similarity between clusters/embeddings/vectors [6]. Among these measures, Cosine Similarity was used in the research rather commonly, which is also used in this study.

2.4.1. Cosine Similarity

The mathematical equation of Cosine similarity between two non-zero vectors A and B is shown in Equation 2.1. When Cosine Similarity is applied, two identical vectors will result in 1, and two orthogonal vectors will result in 0. In the context of text matching, in order to calculate Cosine similarity, text must be converted to vectors. These vectors can be calculated through the term frequency, or word embedding vectors can be used. Word embeddings

have proven to be a better choice for getting semantic information about the text [21]. Since the embedding vectors are normalized, the cosine similarity of embedding vectors is ranged between $[0, 1]$, as in it is equivalent to Euclidean distance. [3]

$$cosinesimilarity = \cos(\theta) = \cos(\mathbf{A}, \mathbf{B}) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \sqrt{\sum_{i=1}^{n} (B_i)^2}} \tag{2.1}$$

2.4.2. Soft Cosine Similarity

One drawback of the Cosine similarity measure is that all the features are assumed to be independent. For the document similarity task, this assumption does not apply. Given two sentences:

1. The children were tired of talking.
2. The kids were exhausted from chatting.

When the stop words are removed, no other words in the sentences are the same. However, the semantic meaning of the sentences is very close to each other, as texts can be semantically related even with no words in common [64].

Soft Cosine similarity [7] is proposed as a solution to tackle this problem. It is an improved version of Cosine Similarity, which considers the semantic similarity of the words by using word embeddings. Equation 2.2 is different from Equation 2.1 with the addition of the matrix S, which is denoted as the relation matrix for words. If the words are independent of each other, the relation matrix becomes the identity matrix; hence the equation becomes equal to Equation 2.1.

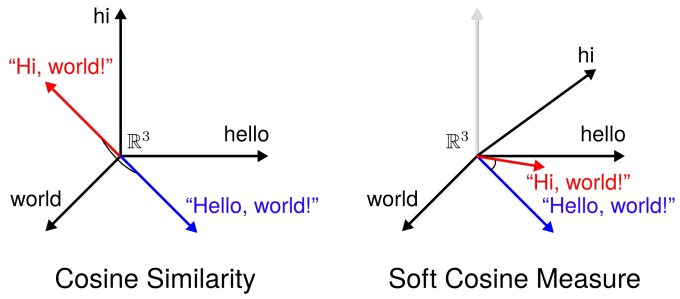


Figure 2.2.: Soft Cosine Similarity takes into account the semantic relations between words, while Cosine similarity assumes the words are independent of each other [4]

$$softcos_S(\mathbf{A}, \mathbf{B}) = \frac{\sum_{i,j=1}^{n} s_{ij} A_i B_i}{\sqrt{\sum_{i,j=1}^{n} s_{ij} (A_i)^2} \sqrt{\sum_{i,j=1}^{n} s_{ij} (B_i)^2}} \tag{2.2}$$

[3] https://en.wikipedia.org/wiki/Cosine_similarity
[4] https://github.com/RaRe-Technologies/gensim/blob/develop/docs/notebooks/soft_cosine_tutorial.ipynb

2.5. Customer Feedback Research

Customer feedback has become a valuable source of improvement for businesses. For many companies, analyzing the trends and insights from the customers has opened up the path to growth in revenue. Until the 2000s, the collection of customer feedback was conveyed mainly through conducting surveys [65]. However, in the past decades, feedback collection processes have evolved with the technological advancements of the digital age. Businesses came up with automatized review systems that collect feedback from customers at a rapid speed. This digitalization of fast-growing unstructured data collection has led research on customer feedback datasets to be conducted more often than ever. The researchers [66] state that the human factor is fundamental to gaining insights from the collected customer feedback. Since the needs of the business problem might require different solutions, applying gained insights on business-level decisions, and figuring out which algorithm is better to use require a human element. Hence, most of the research conducted on the area of customer satisfaction has the basis of (1) modeling the data-generating process by applying theories from economics and consumer behavior, (2) using the customer feedback data as an input to a decision-making model supported by computer science advancements [65].

The most common source of customer feedback is the reviews collected through platforms such as Twitter, Yelp, and Amazon, also through blogs and forums in real-time. However, researchers question whether online reviews accurately reflect general opinions or are impacted by reviews of other customers [65]. Many online reviews include fake reviews [67] and bias, which could cause a market decline. However, online reviews have the advantage of being open and available to everyone, which helps researchers save time in preparing a survey and conducting interviews one by one. Survey-based customer feedback collection is still an ongoing procedure by many businesses, which requires a structured outline to be presented to participants [68]; however, surveys usually have low response rates, plus they tend to get outdated with the applied innovations on the products [65].

Many studies in this area focus on opinion mining, which is the process of finding user opinions about a topic, product, or problem [69]. For this process, sentiment analysis is often used to determine customer satisfaction. Methods such as Naïve Bayes classifier [70], decision trees [71], SVM [72], K-nearest neighbors [73], and logistic regression [74] are methods that have commonly been applied for opinion mining through sentiment analysis. Another way to dive deep into customer feedback is through topic modeling. The topic models have been used to analyze how customers perceive the products. LDA has been broadly used to mine customer opinions through online reviews written on social media platforms [75][76]. In recent years embedding-based topic modeling approaches have been started to apply for customer feedback datasets [49], which gives the performance of topic modeling applications on customer feedback datasets to provide improved results in the future.

3. Data

In this chapter, the datasets used in this work are explained. The exact data cannot be disclosed for both datasets explained below, due to confidentiality policies. The data used for this study is part of the quality data at BMW.

3.1. Welcome Call Dataset

BMW Welcome Call dataset consists of customer feedback about the newly purchased vehicles. BMW customer satisfaction employees log the feedback, and the transcripts are sent to the BMW quality control departments and later on inspected one by one and categorized by hand. The customer feedback is collected through the USA market, hence the samples of the dataset are in English.

The dataset itself has over 100 columns, most of which are not relevant to our study. We have filtered the dataset and created a subset that includes the relevant attributes and samples to our research. The filtered attributes of the data are described in Table 3.1.

Column Name	Description
van_17	The unique anonymized vehicle identifier. (text)
call_date	The date of the call. (timestamp)
category	Hand-labeled category of the customer feedback. (text)
customer_feedback	The recorded customer feedback text. (text)
feedback_type	Labeled type of feedback, 4 categories available: Likes, Difficult to Use, Wants, Defects. (categorical)
production_series	The model name during the production. (text)
model	The commercial model name of the vehicle. (text)

Table 3.1.: Metadata of BMW Welcome Call data subset

3.1.1. Dataset Analytics

498.505 samples have been collected from the Welcome Call dataset. The dataset consists of 62 production models and 131 commercial model names, with the earliest recorded call date of 25.09.2017.

The dataset has no empty entries on the customer_feedback column. The longest entry is 2037 characters long and has 323 words, whereas the entry with the longest words has 375

words with a length of 1997 characters. The shortest one has a length of 1, which is just a dot in order to make the entry not empty.

The dataset is hand-categorized into 253 categories, for example, "Technology", "Interior", "Exterior", "Door panel", and "Apple CarPlay" can be given as examples. Some of these categories can have only 1 entry belonging to them, or the categories can be too similar to each other, which makes them harder to inspect. Hence, it is hard to consider this column as the label for the dataset entries.

The analysis of feedback type, which can be considered as the sentiment of the text, showed that 52.5% of the customers liked the vehicle they purchased. 23.9% reported they are having difficulty using the vehicle, while 13% stated they would have wanted additional features. Only 10.6% of the customers reported defects in the vehicle they purchased. Figure 3.1 shows the feedback type analysis.

For this study, we have chosen not to include the data that has the feedback type "Likes" since the aim of this study is to find issues occurring within vehicles. That left us with 47.5% of the collected samples, which is equal to 236.878.

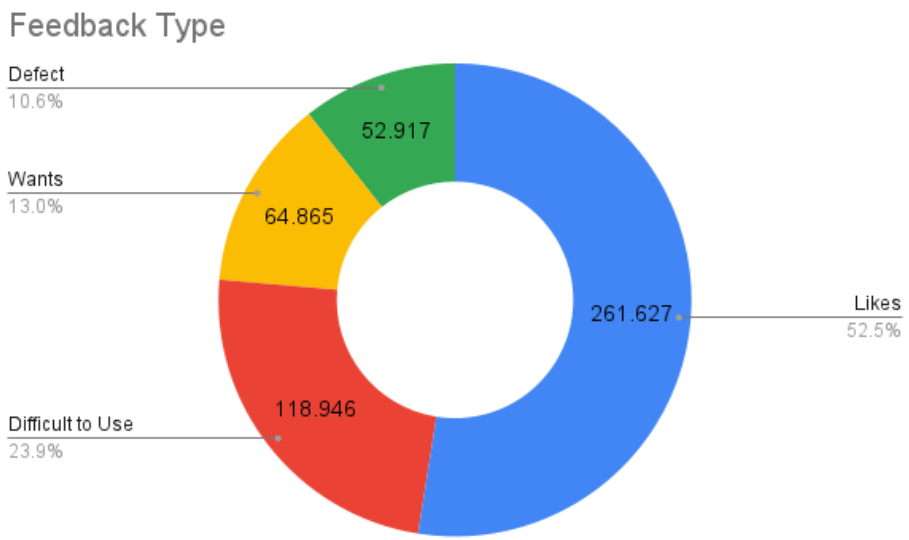


Figure 3.1.: Feedback type

3.2. Problem Quality Management (PQM) Datasets

Problem Quality Management Process consists of two datasets, Incident data and Problem data, as explained in subsection 1.2.2. For our study, we aimed to use samples that are only in English since Welcome Call data does not have any German samples. However, PQM data is not labeled with language information. For this reason, we decided to apply a join between the Incident dataset and the Problem dataset and use language classification to filter

the English samples occurring within the data. 851.932 samples have been collected from the PQM Incident dataset, with 147.378 distinct problem numbers. These problem numbers are used to map PQM Incident dataset to the PQM Problem dataset. By doing so we have collected Problem Descriptions and Problem Titles and created a joined table. The attributes of the created dataset are described in Table 3.2.

Column Name	Description
pqm_incident_number	The unique incident identifier. (numeric)
pqm_problem_number	The unique problem identifier. (numeric)
pqm_problem_assignment_date	The date of incident assignment. (timestamp)
pqm_incident_title	The title of the PQM incident. (text)
pqm_incident_description	The description of the PQM incident. (text)
pqm_model_range	The model name during the production. (text)
pqm_problem_title	The title of the PQM problem. (text)
pqm_problem_description	The description of the PQM problem. (text)

Table 3.2.: Metadata of BMW Problem Quality Management data subset

3.2.1. Dataset Analytics

Both the PQM Incident dataset and the PQM Problem dataset have 100 columns. The earliest entry on the dataset is entered on 09.07.1997, and new data is updated every week. Depending on the production plant, the data entries can be in German or in English. 231 production models exist in the data.

The longest entry in the dataset has a length of 4300 characters with 549 many words. However, the entry with the most number of words has 745 words with 3852 characters. The shortest entry is 1 character long.

3.2.2. Language Identification

One challenge with the PQM dataset was that the PQM incidents could be either German or English. Sometimes the problem title would be in English, but the incidents related to it would still be in German. This is dependent on the production plant; if the vehicle is produced in an English-speaking plant, the data would most likely be English; if it is produced in a German-speaking plant, it could be either German or English. Since PQM problems are generated from a collection of PQM incidents, the language of the PQM Problem would differ based on the location of the creator of the problem. This means that the PQM problem matching a PQM incident can have a different language than the given incident. Since Welcome Call data is in English, we wanted both datasets to have matching languages. To tackle this issue, language identification of the text was important.

We used the PySpark implementation of *FastText* library to classify the language of the text. PySpark was chosen over pandas to store the dataset due to the large number of samples (851K) of the PQM dataset. *FastText* has two models, a smaller compressed version (.ftz) with

917 kB in size and a larger one (.bin) with 126 MB in size. .bin model is stated to report better results compared to the .ftz model due to the difference in the model size. Both models are trained in the same dataset, and able to recognize 176 languages [23] [24].

The experiments, shown in Table 3.3, were firstly done by using both models on *pqm_incident_title* and *pqm_problem_title* columns. The titles were preferred for the language identification experiments due to the shortness of the text and having more clean text compared to the descriptions. After the labeling was done, datasets were filtered to keep samples labeled as English. However, initially, the results were not as good as expected. With the .ftz model, some samples were labeled as Dutch or French, although they were clearly English. With the .bin model, the number of rows labeled as English was more than with the .ftz model. Since the results were noisy, an additional experiment with the .bin model on *pqm_incident_description* column was carried out. For this additional experiment .bin model was preferred since the number of mislabeled samples was less compared to the .ftz model. *pqm_incident_description* was selected since *pqm_problem_description* column is a more general column and it was observed to be in German more frequently than the *pqm_incident_description* column.

	FastText Model	**Text column name**	**Number of samples labeled as English**
1	.ftz	pqm_incident_title	302196
2	.ftz	pqm_problem_title	305283
3	.bin	pqm_incident_title	331397
4	.bin	pqm_problem_title	325294
5	.bin	pqm_incident_description	298178

Table 3.3.: Language identification with FastText for BMW Problem Quality Management data subset. Highlighted rows are used for the inner join between datasets.

The experiments showed that some samples had *pqm_problem_title* and *pqm_problem_description* columns in German, although *pqm_incident_title* and *pqm_incident_description* columns were in English. Hence we decided to apply an inner join to the resulted subsets of experiments (1) and (5) highlighted at Table 3.3 on the *pqm_problem_number* column, due to them having less number of rows compared to other experiments, which resulted in a subset of 247.608 English samples. Inspecting the resulted subset showed that the *pqm_incident_description* column has indeed consisted of English samples with the exception of a few noisy samples that have both English and German sentences.

3.3. Preprocessing

In this section, we describe our general data preprocessing pipeline, which is applied to both of the datasets mentioned above during the experiments. Statistical topic models especially require preprocessing to generate meaningful results, whereas embedding-based models still perform adequately without essential preprocessing steps. Due to this difference between requirements for preprocessing, we decided to design a preprocessing pipeline that

works for text columns of both datasets and with the option to select the steps as boolean parameters.

During data analysis, we observed the Welcome Call dataset contains a lot of contractions and stop words, whereas the PQM dataset contains a lot of digits and non-alphabetical characters as part of error coding.

For our preprocessing pipeline, we have applied the following steps, also shown in Figure 3.2:

- Expand contractions (don't ⇒ do not)
  Python library *contractions* [1] is used to expand the contractions for this study.
- Replace newline and tab characters (Writer: \n message \n Customer: \t ⇒ Writer: message Customer:)
- Remove apostrophes (new entry 'Likes' ⇒ new entry Likes)
- Remove digits and non-alphabetical characters (Dated 10.08.2010 ⇒ Dated)
- Lower case (CUsTomer ⇒ customer)
- Lemmatize verbs
- Stop word removal
- Lemmatize nouns
- Remove entries with empty text values from data:
  As a result of the preprocessing, some of the entries in the datasets resulted in being empty. We have removed these empty entries as the final step.

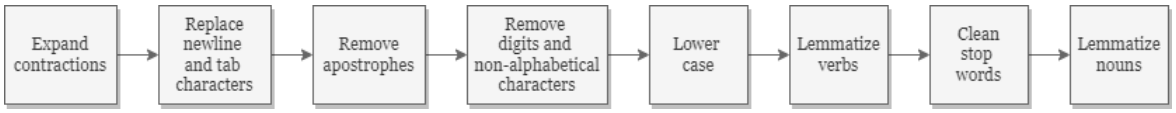


Figure 3.2.: General preprocessing pipeline

These steps are quite straightforward, except for stop-word removal and lemmatization steps, which are explained in more detail below. For the stop-word removal and lemmatization, we used Welcome Call dataset analysis and then applied the same steps to the PQM dataset to see if it is applicable.

3.3.1. Stop-word Removal

We used Python library *spaCy* [77] to clean English stop words from the samples. *spaCy* allows custom stop words to be added to the default stop words, which is a feature we

[1] https://github.com/kootenpv/contractions

leveraged. The decision for the design of the stop-word removal step was based on our data analysis and decided by the word frequency descriptions.

The word frequency distribution of Welcome Call data is shown in Figure 3.3. Some of the most frequent words in the dataset is shown to be *'the'*, *'to'*, *'and'*, *'that'* which are stop words, and also words such as *'customer'*, *'stated'* and *'writer'* seem to occur frequently.

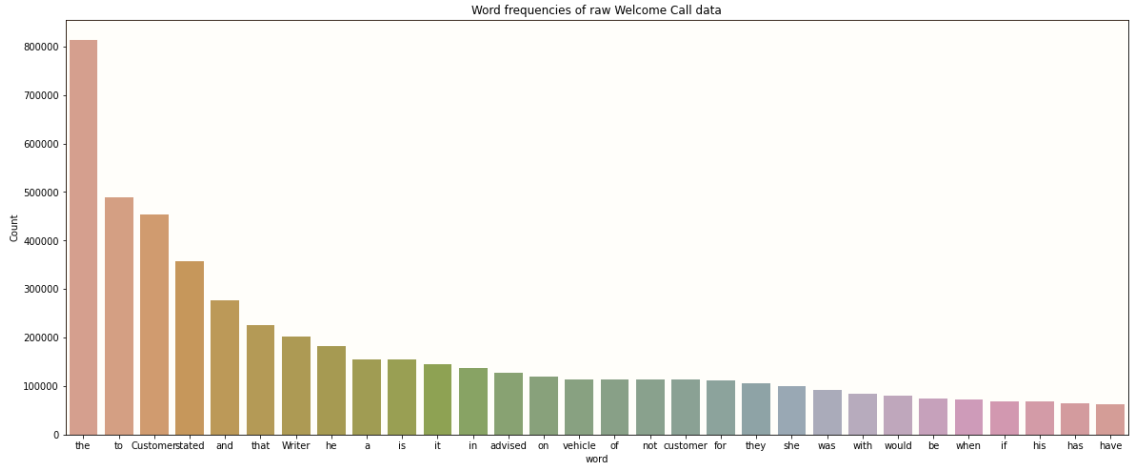


Figure 3.3.: Word frequency of raw Welcome Call dataset

After applying stop word removal with spaCy's default stop word list, the most frequent words in the Welcome Call dataset became *'customer'*, *'stated'*, *'writer'*, *'vehicle'* and *'advised'*, as shown in Figure 3.4. However, these words are observed to be occurring much more often than the rest of the frequent words, which indicates that they are not adding any significantly important value to the data. The word *'customer'* occurs in the data more than 50.000 times, while the sixth most common word *'like'* occurs less than 10.000 times. This led us to the decision of removing the words *'customer'*, *'writer'*, *'state'* and *'advise'* from the dataset, the word frequency results with this custom setup of stop word removal is shown in Figure 3.5.

3.3.2. Lemmatization

Conventionally, lemmatization or stemming is applied to the dataset after stop-word removal. One downside we observed by applying stop word removal first is that the correct forms of the stop words had to be in the list in order to remove all possible forms from the dataset. Default stop word lists of text preprocessing libraries usually contain all possible forms for the stop words; however, since we wanted to add custom words to the stop word list, it turned out to be challenging to add every single form of the additional custom words. For example, the word *'state'* has been removed from the dataset after the stop word removal step; however, *'stated'* and *'states'* still seemed to be some of the most frequent words occurring at this point. Our solution to automate this was through converting the verbs to their root form and then applying the stop word removal to the dataset.

Given the differences between stemming and lemmatization explained in subsubsec-

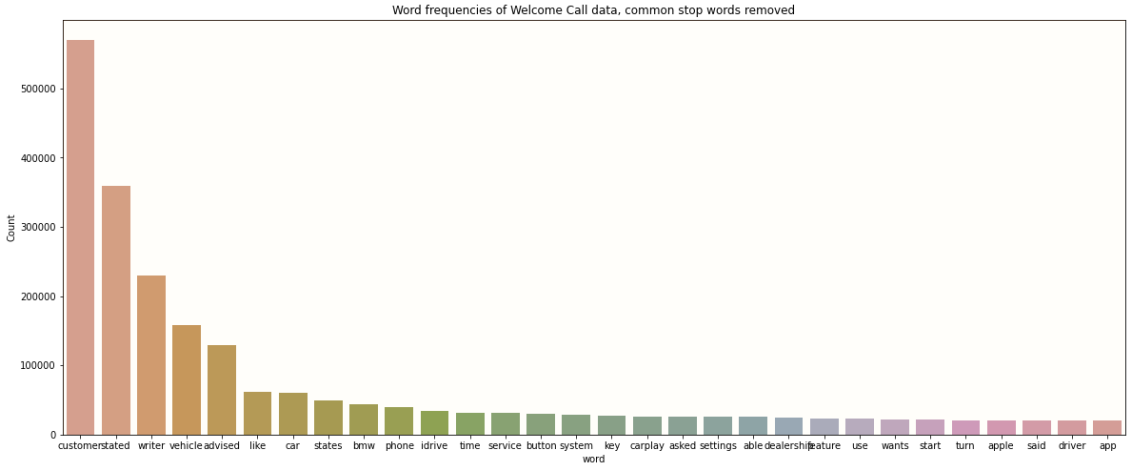


Figure 3.4.: Word frequency of Welcome Call dataset, common stop words removed

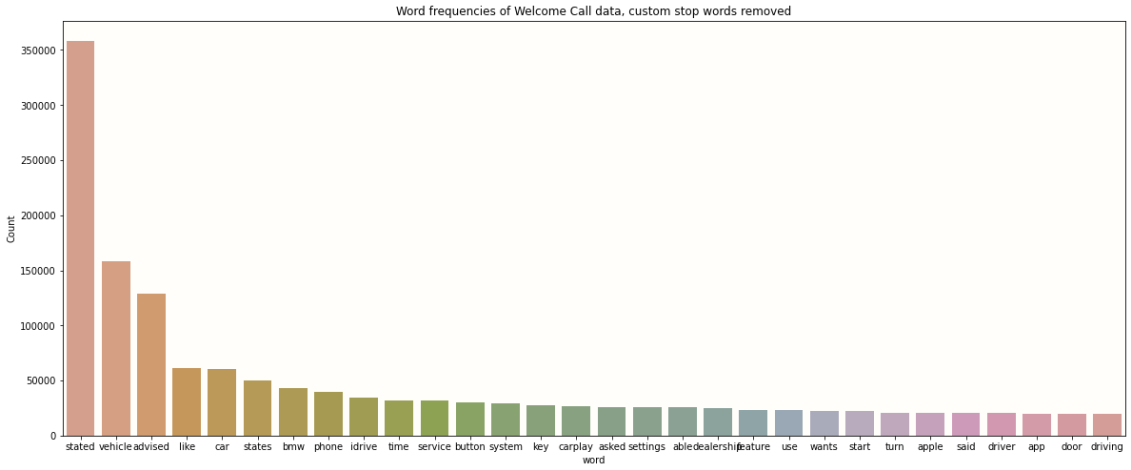


Figure 3.5.: Word frequency of Welcome Call dataset, custom stop words removed

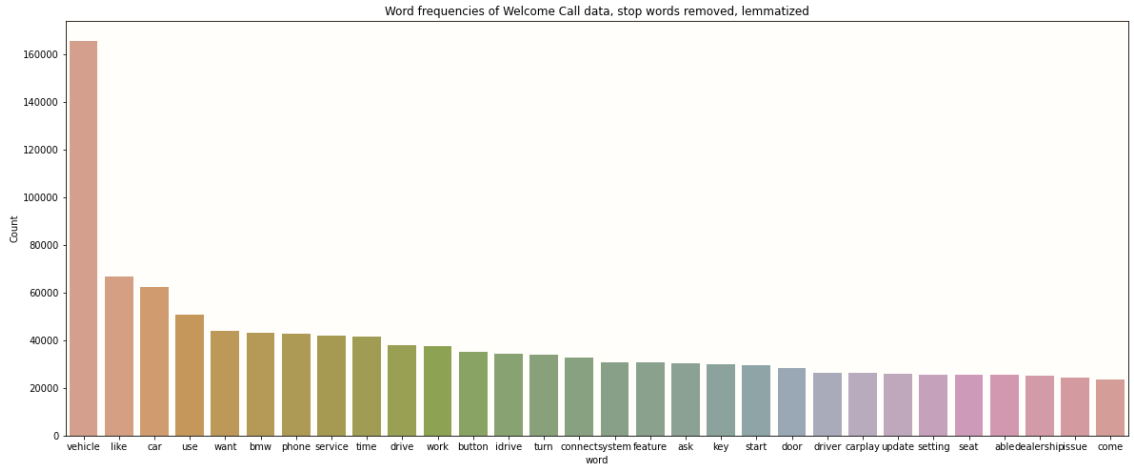


Figure 3.6.: Word frequency of Welcome Call dataset, lemmatized and stop words removed

tion 2.1.2, we chose to use a lemmatizer instead of a stemmer for the text normalization part of our preprocessing step. For this purpose, we used the Python library Natural Language Toolkit (NLTK) [78], which is a text preprocessing library that has many useful features. We used the WordNet Lemmatizer of NLTK to lemmatize the words. By default, WordNet Lemmatizer uses *noun* as the part-of-speech tag to be cleaned, however the part-of-speech tag parameter can be changed to *verb*, *adjective* or *adverb*. We have observed that applying verb lemmatization before noun lemmatization resulted better for the given datasets. Figure 3.6 shows the word frequencies of the Welcome Call dataset after preprocessing steps are applied. As can be observed from the plot, the most common words are converted to their base form.

3.3.3. Preprocessing of the PQM dataset

Applying the same steps to the PQM dataset showed that this dataset does not have significant words to be removed as the Welcome Call dataset has, so we did not remove any additional custom words from the PQM dataset as part of stop-word removal. Figure 3.7 shows the word frequencies of raw PQM data, which shows that the stop words are the most frequently occurring words. Figure 3.8 shows the word frequencies of preprocessed PQM data, with no additional words removed. It can be observed that after the preprocessing, there is no word that is significantly more frequent than the rest of the words. The most frequent word in the cleaned dataset is '*find*' with roughly 7000 occurrences.

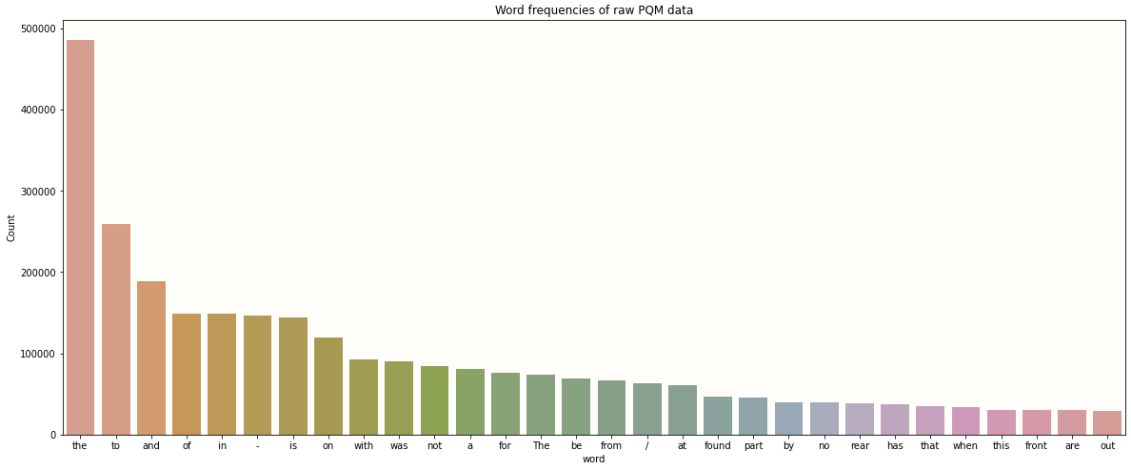


Figure 3.7.: Word frequency of raw PQM dataset

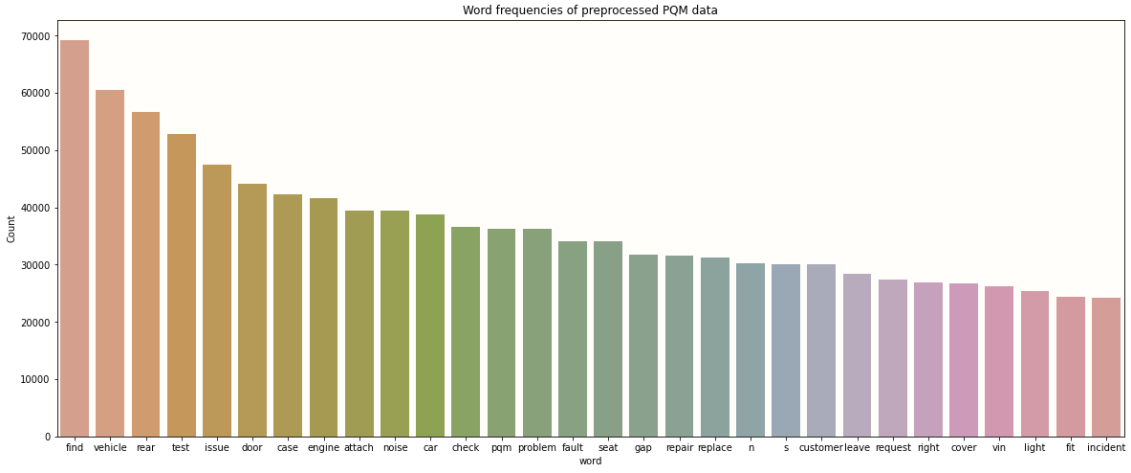


Figure 3.8.: Word frequency of preprocessed PQM dataset

4. Methodology

This study consists of several pipelines: Preprocessing, Topic modeling, and Topic matching. The preprocessing pipeline is previously explained in section 3.3 and is applied to both Welcome Call and PQM datasets. In this chapter, we explain our methodology for topic modeling and topic matching approaches, as well as the evaluation metrics and methods used in this study. Figure 4.1 displays the block diagram of our research methodology and the processes.

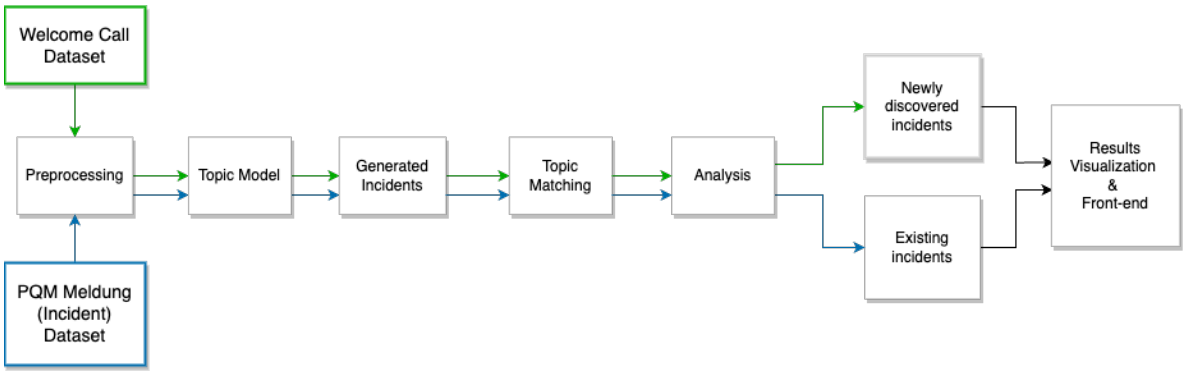


Figure 4.1.: Block diagram of the methodology of this study. The steps are applied separately to both datasets and are shown with green arrows for the Welcome Call dataset and blue arrows for the PQM dataset. First, they are preprocessed for model training. A topic model is used to extract the topics, which are the 'generated incident' clusters. Then these topics are matched together by applying text similarity measures. The output of these steps is a list of topic pairs, which is analyzed to determine the new incidents occurring in the Welcome Call dataset and to confirm existing incidents which are 'known problems' from the PQM dataset. The final step of this pipeline is the visualization of the findings of topic modeling and topic matching steps.

4.1. Topic Modeling

One of the research aims of this study is to find out which topic modeling approach would be better for retrieving information from customer feedback datasets. We used two methods for topic modeling to discover the topics occurring in the datasets, LDA as the traditional approach and BERTopic as the embedding-based approach, which are explained previously

in more detail on section 2.3. We have set up two pipelines for each topic modeling approach to train the models and obtain results. The pipelines are applied to both the Welcome Call dataset and the PQM dataset. The decision on which topic model provides a better representation is carried out by inspecting the intertopic distance maps and the generated topic words and applying topic coherence measures. For the topic modeling task, it is crucial to figure out the correct number of topic clusters to get a better representation of the latent space. A grid search is applied to determine the correct hyperparameter settings for each parameter, and the necessary visualizations and outputs are recorded.

4.1.1. Traditional Approach

LDA[2] is used as the baseline to compare the results of our research due to its popularity in topic modeling research, as explained in section 2.3. This model provides researchers with an explicit representation of the given documents [48]. Given an input of documents, LDA splits them into topics and topic words. LDA is a statistical model, so it does not have the information on semantics or context as word embedding-based models have. That is why it is crucial to preprocess the data before applying this method. For this reason, we applied our preprocessing pipeline explained in section 3.3 to both datasets.

4.1.2. Embedding-based Approach

As the embedding-based approach for topic modeling, we used BERTopic [10], which uses an embedding model to generate document embeddings. It then reduces the dimensionality of the gathered embeddings with UMAP and applies HDBSCAN to find semantically similar document clusters. Later, it uses class-based TF-IDF to generate topic word representations. This method allows auto-generating the number of topics based on predefined minimum cluster size, so it does not require a specific number of topics to be defined as a parameter.

4.2. Topic Matching

After clustering documents and extracting the topic words, this study aims to find out how similar the generated topics of both datasets are. For this purpose, we plan to experiment and compare the results of text similarity measures Cosine Similarity and Soft Cosine Similarity. As explained in section 2.4, Cosine Similarity calculates the distance between two vectors, whereas Soft Cosine Similarity uses fixed-length word embeddings as an addition to the Cosine Similarity formula to consider the contextual information when calculating the similarity between two sequences.

Our approach is first to determine the better-resulting topic modeling method, then compare the topic-level similarities of the Welcome Call dataset and the PQM dataset generated by the selected approach. If LDA provides better topic representations, Cosine Similarity will be applied using TF-IDF representations on topic words for each model, which will provide a comparison between count-based method TF-IDF and fixed-length Word Embedding method. BERTopic uses contextual embeddings to generate document embeddings and later reduces

dimensionality to output topic embeddings. Similar documents are expected to have similar vectors due to containing similar words; hence using topic embeddings as input for similarity measures is expected to be useful for topic matching. So, if BERTopic achieves better topic representations, Cosine Similarity will be applied directly to generated topic embeddings instead of converting the topic words to TF-IDF format, and the comparison of similarity measures will focus on fixed-length embeddings and contextual embeddings.

4.3. Evaluation Metrics

This section describes the evaluation metrics and measures used in this study. Several evaluation metrics have been reported as this study works through two subproblems: topic modeling and topic matching.

4.3.1. Evaluation of Topic Models

Researchers have developed and used many metrics to evaluate topic models, although there is no consensus on which evaluation method is the standard way for topic model evaluation, unlike other NLP tasks [48]. Besides, since topic models depend on the hidden structures in datasets, it often requires understanding the related concepts, and none of the standardized metrics have managed to achieve that so far. Although reporting analytical metrics is still commonly applied in topic model evaluation, human judgment is still vital when it comes to the evaluation of topic models. Hence for this study, we have used both analytical measures and human evaluation.

Analytical Measures

The most popular analytical measure reported by studies is the perplexity score, an intrinsic evaluation method that measures how well a probability distribution or probability model predicts a sample [15]. However, over time the researchers have shown that perplexity negatively correlates with human judgment when it comes to topic coherence [79][80]. That is why we used coherence metrics for this study as the analytical measures instead of the perplexity score.

Topic coherence has been proposed as an intrinsic evaluation method for topic models. The most commonly reported coherence metrics are *Umass*[81], *C_v*[82], *UCI*[83] and *NPMI*[84]. Among these, *C_v*, *UCI*, and *NPMI* use the sliding window approach, which makes the computations take a long time. Among the mentioned metrics, *Umass* and *C_v* are the most popular ones; however, *C_v* is not recommended to be used due to the replication of the results using this metric can be faulty[1], and with small probabilities, it does not yield stable results. We have reported the *Umass* coherence metric, given these reasons. The higher the reported *Umass* score, the better the topic coherence. This score measures how much, within

[1]https://github.com/dice-group/Palmetto/issues/13

the words that describe a topic, on average, a common word is a good predictor for a less common word.[2]

$$score_{UMass}(w_i, w_j) = log\frac{D(w_i, w_j) + 1}{D(w_i)} \tag{4.1}$$

Equation 4.1 shows the equation for *Umass*, where $D(w_i, w_j)$ counts the number of documents containing the words w_i and w_j and $D(w_i)$ counts the number of documents containing words w_i. The score function is not symmetric, so the order of words matters for calculations. The score calculates how often two words, w_i and w_j, appear together in the corpus [85].

Human Evaluation

Even though there are standardized metrics for topic modeling, it is stated by the studies that the metrics are not always reliable and do not reflect human judgment accurately [80]. Most of the topic modeling implementations don't have objective evaluation metrics [48][54]. Hence, human evaluation is essential to evaluate topic models, especially in a domain-specific study. Some of the proposed approaches of human metrics for topic coherence are intrusion and rating [79]. The idea behind the intrusion task is to add one irrelevant word between the topic words and ask the user to identify which word does not belong [80]. The rating task provides the raters with a scale and directly asks if the topic words make sense [79]. The generated topic words are used to evaluate the topic models in both cases. The topic words can be displayed in raw form or visualized through various visualizations. Intertopic distance maps and Word Clouds are used in this study as visualizations to evaluate the models with human judgment.

- **Intertopic Distance Maps**
  An intertopic distance map is a visualization of the topics in a two-dimensional space. Each topic is represented as a circle cluster. The size of the area of each circle is determined by how many documents belong to that topic. Different libraries are used for each topic modeling method to generate intertopic distance maps.

- **Word Clouds**
  A 'Word Cloud' is a visual representation of words proportional to their frequency [86]. Word clouds are a simple visualization method and frequently used in topic modeling applications, which allow the words with higher frequency to appear larger, hence attracting the focus on more relevant words for the generated topic. It is helpful to analyze the generated topic words quicker in user studies for evaluating topic models.

4.3.2. Evaluation of Topic Matching

The topic matching task of this study uses text similarity measures, which is essentially a binary classification problem of whether the matched pair is similar or dissimilar. Previous

[2]http://qpleple.com/topic-coherence-to-evaluate-topic-models/

research [87][88] has shown that to evaluate the performance of text similarity measures precision, recall, and F1 score have been used to evaluate whether the matched pairs are truly similar or not. The decision on which samples are correctly predicted is usually determined by specifying a threshold for similar pairs [88] on previously paired sentence datasets [87]. All three of the measures range between 0 and 1.

Precision

Precision is the ratio of correct predictions to the total instances that were predicted as positive.

$$precision = \frac{true\_positives}{true\_positives + false\_positives} \tag{4.2}$$

Recall

Recall is the ratio of correct predictions to the total of positive instances, including the ones that were missed by the classifier, that gives a general idea about the system's sensitivity.

$$recall = \frac{true\_positives}{true\_positives + false\_negatives} \tag{4.3}$$

F1 Score

The F1 score is the harmonic mean of precision and recall. Presumably, a good model should have high precision and high recall, which is not the case in most real-world applications. F1 score gives equal importance to precision and recall; hence F1 score will be higher if both precision and recall are high. If there are multiple classes, the F1 score is macro averaged.

$$F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{4.4}$$

5. Experiments and Results

This chapter discusses the experimental settings for this study, whose methodology has been explained under chapter 4. We first explain each experiment's setup, followed by the results. The experiments aim to try as many different setups as possible to compare the results and evaluate them later.

5.1. Topic Modeling

We have used two topic modeling techniques for our experiments on topic modeling: LDA as the traditional approach and BERTopic as the embedding-based approach. We have set up two pipelines to train the models and obtain results. We have worked with different parameters for data preprocessing and model training for each pipeline.

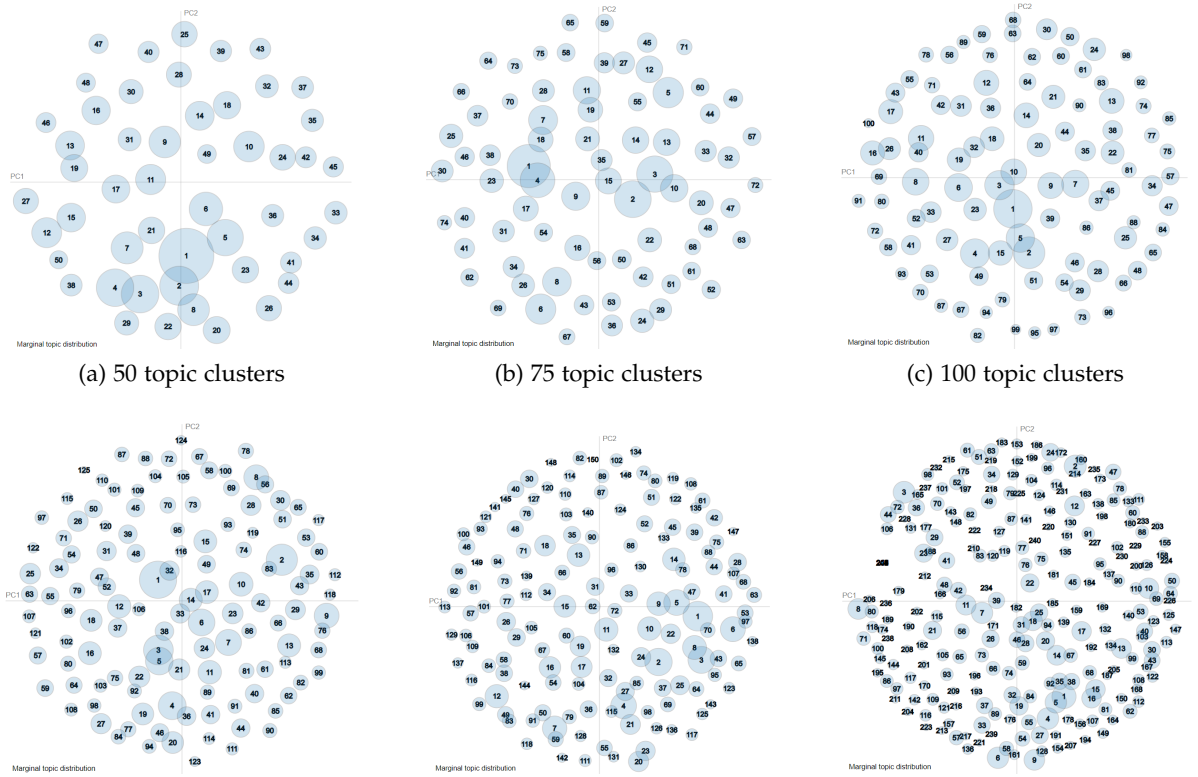
5.1.1. Latent Dirichlet Allocation (LDA)

For the implementation of LDA, gensim[89] library is used due to its support for coherence metric implementation, which is later leveraged for the results. The number of topics for the LDA model has to be determined beforehand. As we have no prior knowledge of the latent space of datasets, we applied a grid search for the number of topic clusters hyperparameter ranging between 50 and 150, with a step size of 25. We have also added the edge case 250 topic clusters to our search space. [48] state that the most significant parameter to be tuned of LDA is the number of topic clusters, which is why we focused on optimizing this parameter.

We first inspected the intertopic distance maps of the different number of topic clusters; for this purpose, PyLDAVis [90] is used, which is a Python library that uses the output of LDA to provide visualizations of topic clusters. It shows the intertopic distance map generated by the model output and the most-probable words associated with each topic cluster. After inspecting the intertopic distance maps, we looked into the quality of topic words per model to interpret the results. Topic words are an essential indicator to determine whether the generated topic is meaningful or not. Each topic of LDA consists of 30 words, of which the word probabilities sum up to 1. Among the 30 words, some might have probabilities very close to zero; this is not a parameter that can be fine-tuned in the gensim implementation of LDA. Topic words can also be represented in Word Cloud format. After this inspection, we tried to find an optimal setting by narrowing down the search space by searching between the two better-resulting cluster numbers with the step size of 5. This sequence of experiments is tried separately for both Welcome Call and PQM datasets. As evaluation metrics, Coherence scores (*Umass*) are calculated by the gensim library.

Welcome Call Dataset

Figure 5.1 shows the intertopic distance maps for a various number of topic clusters setting. Through the plots, it was observed that over 100 topic clusters are too many, and 50 topic clusters are too few to represent the Welcome Call dataset properly.



(a) 50 topic clusters (b) 75 topic clusters (c) 100 topic clusters

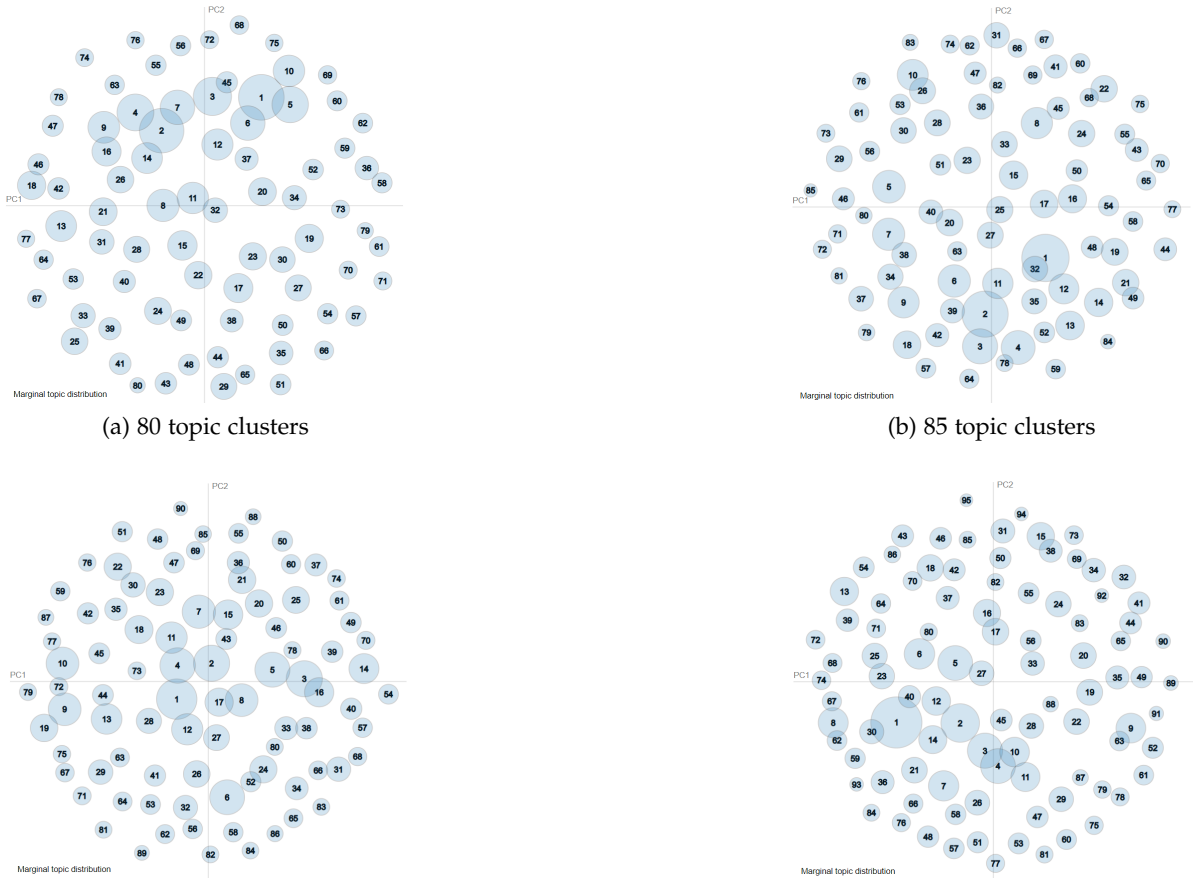
(d) 125 topic clusters (e) 150 topic clusters (f) 250 topic clusters

Figure 5.1.: Comparison of the intertopic distance maps for Welcome Call with LDA

Table A.2 compares topic words related to keywords '*brake*', '*seat*' and '*mirror*', for each number of topic clusters we have experimented with. The lower number of topics, such as 50, clusters many words together and does not show any specification on the topic. The higher number of topics, such as 125, 150, or 250, contains many outlier words, even though some words are relevant to the topic. 250 proves to be too high for topic cluster number selection. Keyword '*seat*' for this cluster number shows that it collects irrelevant words as if they are randomly assigned to the topic; for example, '*preventive*' and '*prevelant*' are consecutive words in the dictionary and they are added to this topic's word list since the conditional word probabilities for the next word have gotten close to zero. Topic cluster numbers 75 and 100 seem to have the most coherent topic word list compared to the others, although they still have outlier words that do not add information to topic definitions.

Given the inspection of intertopic distance maps and the topic words, we have selected the

75-100 range to narrow the search space for the topic clusters. Intertopic distance maps for topic cluster numbers 80, 85, 90 and 95 are shown in Figure 5.2. Unlike the Figure 5.1, the intertopic distance maps for these number of topic clusters do not seem to be much different from each other, which is why we have looked into topic words.



(a) 80 topic clusters

(b) 85 topic clusters

(c) 90 topic clusters

(d) 95 topic clusters

Figure 5.2.: Comparison of the intertopic distance maps for Welcome Call with LDA, 80-95 range

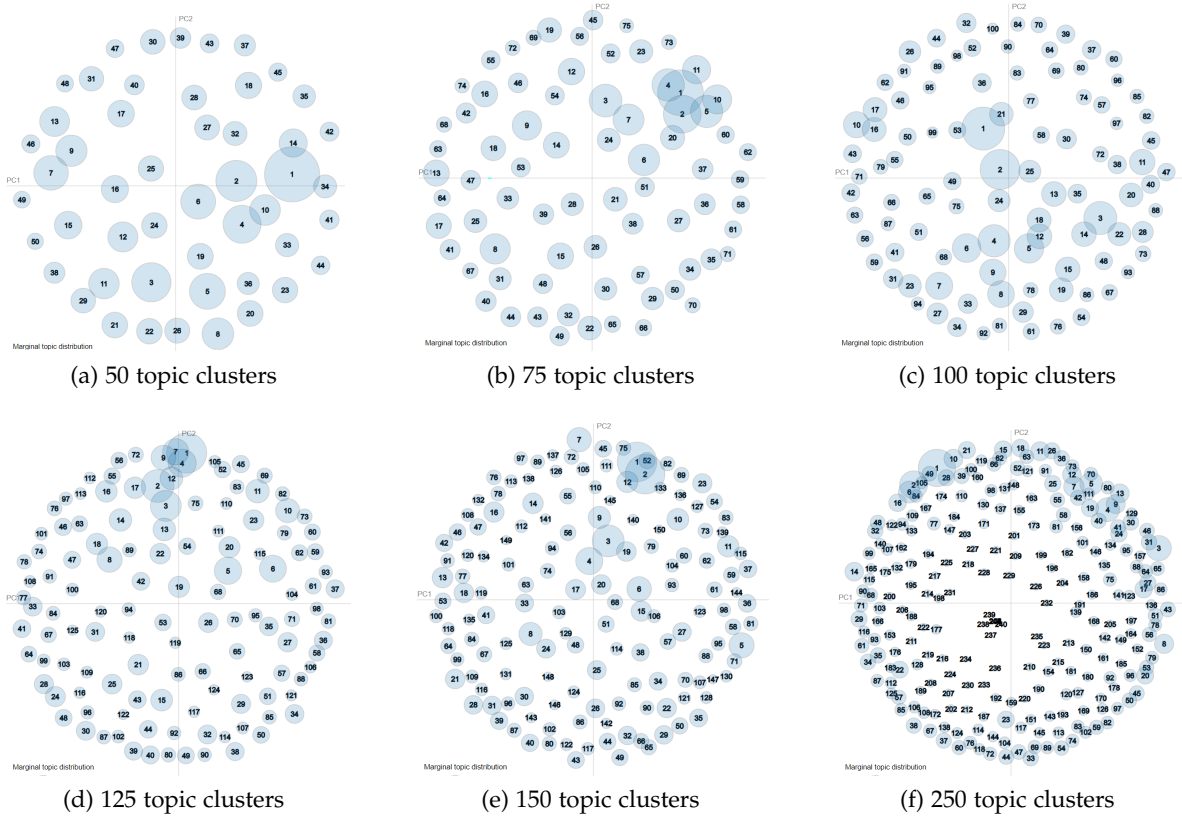
Number of topic clusters	Topic words
80	little, gear, design, wifi, like, shift, bite, expect, small, storage, area, improve, lever, compartment, selector, use, mpg, easy, find, buckle, fill, want, sate, occasionally, place, way, aid, average, car, vehicle
85	driver, wife, let, secondary, primary, document, fresh, smell, sate, output, recommend, value, possibility, know, expire, speak, vehicle, life, daily, think, renew, subscribe, combustion, eliminate, somewhat, ontario, tear, http, allow, like
90	enable, month, default, year, release, free, trial, vehicle, come, want, response, proceed, like, whichever, need, inquire, cooper, jerk, kit, xdrive, tune, period, throttle, perfect, ask, expensive, horsepower, solve, balance, torque
95	away, position, wife, trouble, best, approach, especially, practice, electronic, mybmw, walk, collection, tighter, spouse, manufacturer, use, try, bright, expire, renew, uneven, work, find, strap, firm, proximity, modify, leave, alter, vehicle

Table 5.1.: Examples of bad topic words per topic cluster

Table A.3 shows example topic words with keywords '*brake*', '*seat*' and '*mirror*' which don't give enough to distinguish the quality of topics, since the generated topics for the selected keywords are similar to each other. Hence, we decided to compare some 'bad' topics. Table 5.1 shows some topics that do not seem meaningful to human eyes. All of the models have some topics that are collections of words that do not make sense together; however, LDA statistically clustered these words into a topic since they appear together more often. For example, the 90 topic clusters model has the words '*month*' and '*year*' clustered together, which are related, but this topic does not indicate any possible production issue when the topic words are analyzed.

PQM Dataset

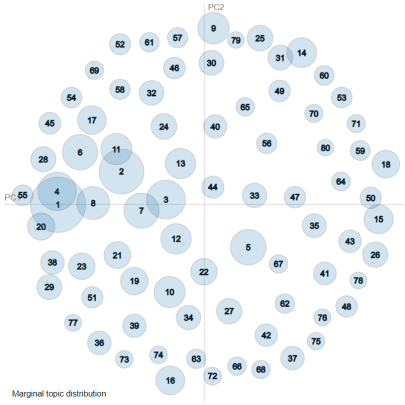
Given the above experimental setting with the Welcome Call dataset, the same steps have been applied to the PQM dataset with the same parameter settings. Figure 5.3 shows the same assumptions observed with the Welcome Call dataset also apply to the PQM dataset. Less than 75 topic clusters are too few to represent the dataset properly, and more than 100 topic clusters are too many; hence we have focused on topic clusters between 75 and 100, which the intertopic distance maps are visualized on Figure 5.4.



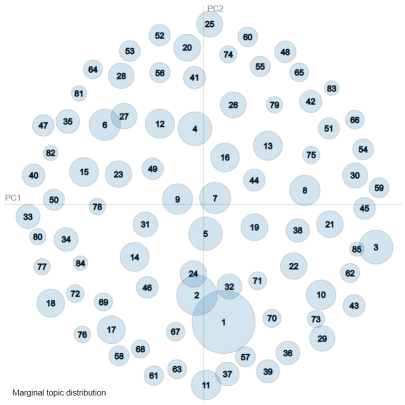
(a) 50 topic clusters (b) 75 topic clusters (c) 100 topic clusters

(d) 125 topic clusters (e) 150 topic clusters (f) 250 topic clusters

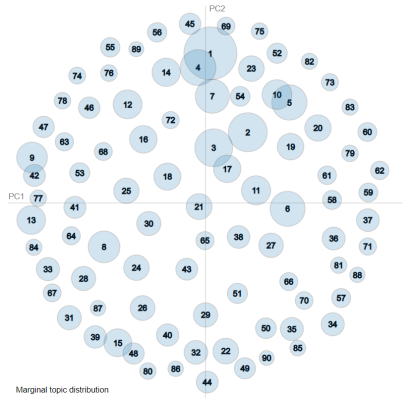
Figure 5.3.: Comparison of the intertopic distance maps for PQM with LDA



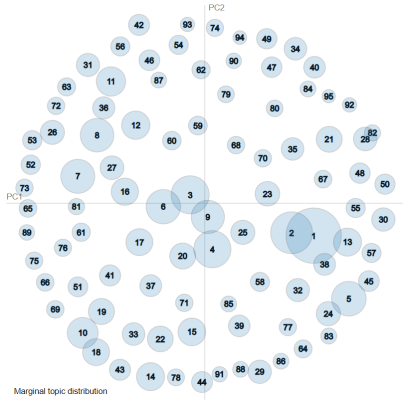
(a) 80 topic clusters



(b) 85 topic clusters



(c) 90 topic clusters



(d) 95 topic clusters

Figure 5.4.: Comparison of the intertopic distance maps for PQM with LDA, 80-95 range

Table A.1 displays example topic words with keywords '*brake*', '*seat*' and '*mirror*' for number of topic clusters 80, 85, 90 and 95. One noticeable detail we observed with the PQM data is that it contains many abbreviations that are frequently used, such as '*lhs*', '*vfc*', '*fzd*', which affects the interpretability of the topic quality negatively, compared to Welcome Call topics. These abbreviations are production-specific terms, which cannot be cleared by stop word removal, nor can they be efficiently removed by hand. In order to determine whether the classified abbreviations make sense or not, expert knowledge is required. Apart from this issue, the topic words shown in the Table A.1 seem to be similar to each other for the given keywords for each cluster number.

Evaluation

Table 5.2 displays *Umass* scores for Welcome Call and PQM datasets with all the topic cluster numbers explained above. The *Umass* scores have a pattern of increasing inversely proportional to the topic cluster numbers, with exceptions of the 85 clusters model for Welcome Call and the 90 clusters model for PQM dataset, as well as the 250 clusters model

for both datasets. The 250 topic clusters model's scores were lower because some of the topics have words with very low probabilities (e.g., $9.814094e-08$), such that the generated words are not relevant to that topic but sampled from the dictionary. This issue has been observed with several topics with almost always the sampling of the same words in the same order, which increases the *Umass* score since this coherence measure calculates how often two words appear together (see subsubsection 4.3.1).

	Umass	
Number of topic clusters	**Welcome Call models**	**PQM models**
50	**-3.612**	**-4.048**
75	-4.584	-4.761
80	-4.906	-4.836
85	-4.837	-5.230
90	-5.152	-5.219
95	-5.193	-5.459
100	-5.501	-5.645
125	-5.591	-6.109
150	-6.167	-6.433
250	-4.811	-5.605

Table 5.2.: LDA model experiment results for Welcome Call and PQM datasets. *Umass* has been reported as the coherence score. The closer the *Umass* score is to zero, the more coherent the model is. The calculations are based on the top 20 topic words.

5.1.2. BERTopic

BERTopic [10] is an embedding-based topic model, which by default uses a Sentence-BERT (SBERT) [32] embedding model 'all-MiniLM-L6-v2' [1], that maps sentences and paragraphs to a 384-dimensional dense vector space. Other embedding models, such as BERT, are also supported; however, Sentence-BERT has yielded better results due to sentence-level embeddings than BERT, which generates word-level embeddings. Sentence-BERT is also advantageous over prior models, as it increases computational performance on tasks such as hierarchical clustering, which takes 5 seconds with SBERT and 65 hours with classical BERT.

BERTopic has several hyperparameters that can be tuned for optimization. The pipeline we designed integrates several of the parameters provided by the library.

- *min_topic_size*: The minimum size of the topic. The default value is 10 documents per topic. Increasing this parameter results in less number of topic clusters.
- *nr_topics*: The number of topics to be generated by the topic model. If not specified, it will generate as many topics as possible.

[1] https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

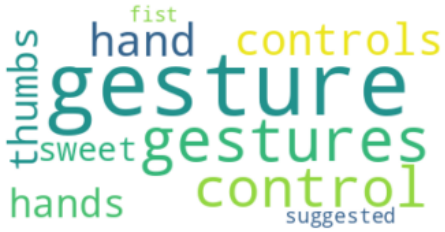
- *top_n_words*: The number of words per topic to extract. The default value is 10, and the maximum allowed is 30.

Raw data vs. Preprocessed data

For the BERTopic model, first, we wanted to explore the model's behavior on BMW datasets. We experimented with raw and preprocessed data without limiting the cluster number parameter and keeping every other parameter in the default setting, except for the minimum topic size parameter, which we fixed to 40. The resulting number of topic clusters for preprocessed and raw settings for both datasets are shown in Table 5.3.

Dataset	Preprocessed or Raw	Number of topic words	Number of documents per cluster	Number of topic clusters
Welcome Call	Raw	10	40	702
	Preprocessed	10	40	547
PQM	Raw	10	40	550
	Preprocessed	10	40	445

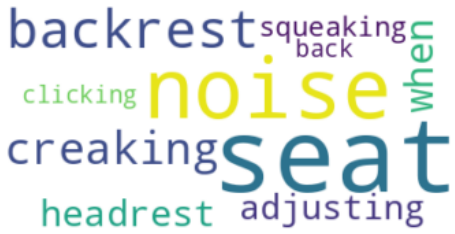
Table 5.3.: Number of topic clusters generated with the specified settings for BERTopic model



(a) Raw topic for Welcome Call

(b) Preprocessed topic for Welcome Call

Figure 5.5.: Comparison of the same topic for raw vs. preprocessed Welcome Call dataset



(a) Raw topic for PQM

(b) Preprocessed topic for PQM

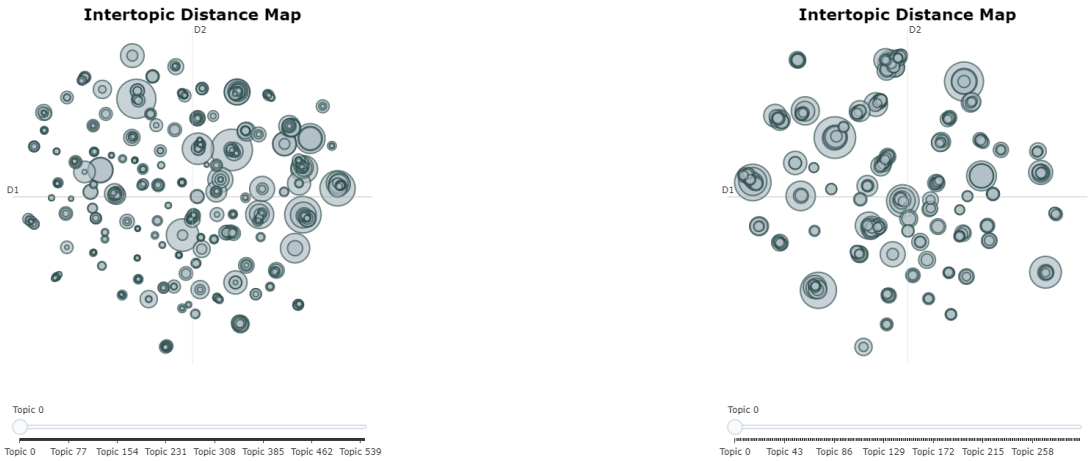
Figure 5.6.: Comparison of the same topic for raw vs. preprocessed PQM dataset

Since BERTopic uses contextual embeddings, we expected the model results to be good even if the data is not preprocessed. However, with the raw data, the generated topic words turned out to be noisy and did not represent the topics in a sensible way. The plural and present tense suffixes were still observant and clustered together with the singular versions and other tenses. Hence, we decided to continue our process with the preprocessed datasets. An example topic comparison for Welcome Call data is shown in Figure 5.5 and for PQM data in Figure 5.6.

Auto-Generated vs. Reduced Topic Cluster Numbers

As the results shown on Table 5.3 indicate, the automatically generated topic numbers were too many to interpret any results. Since the aim of topic modeling is to observe underlying topics in the data, having too many topic clusters made the analysis time-consuming, which contradicts our research aim; for this reason, finding the optimal number of topic clusters proved to be a challenging task.

Leveraging the built-in features of BERTopic, we decided to reduce the number of topics by applying the *reduce_topics* function of BERTopic to our trained models. By selecting a specific number of topics, the model clusters more documents together, which changes the *min_topic_size* parameter. We decided to reduce the number of topic clusters to 300. Since in the previous experiment, the number of clusters for preprocessed datasets was 547 for Welcome Call and 445 for PQM, we decided 300 would be a good enough reduction for the first step. With this reduction, the smallest cluster for Welcome Call contained 169 documents, and for PQM, the smallest cluster had 89 documents.



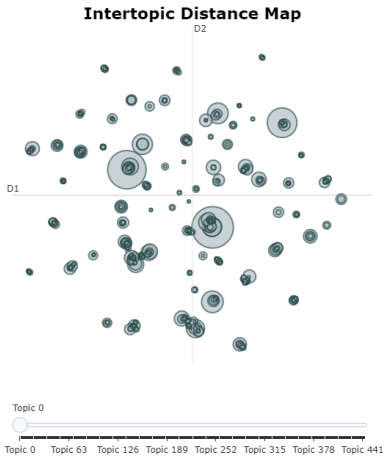
(a) 547 topic clusters for Welcome Call

(b) 300 topic clusters for Welcome Call

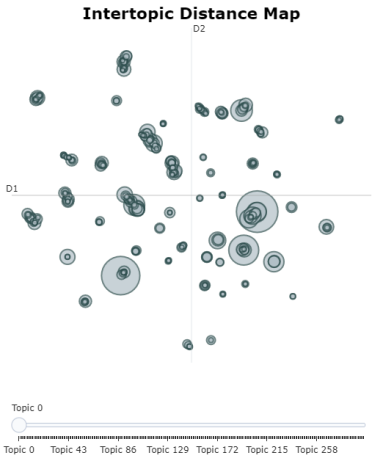
Figure 5.7.: Comparison of the intertopic distance map for Welcome Call auto-generated topics and reduced topics

Comparisons between auto-generated versus reduced topic clusters numbers are shown

as intertopic distance maps on Figure 5.7 for Welcome Call and Figure 5.8 for PQM. After examining the intertopic distance maps of the models, which is a built-in visualization provided in the implementation of the model, we have concluded that even 300 topic clusters are too many to represent data more simply and compactly. Even though the difference in cluster sizes on the graphs seems significantly better after topic reduction, there were still too many overlaps on topic clusters, which means that the defined topics were not that different.



(a) 445 topic clusters for PQM



(b) 300 topic clusters for PQM

Figure 5.8.: Comparison of the intertopic distance map for PQM auto-generated topics and reduced topics

Experiment Setup

After deciding on using the preprocessed data and realizing we need less number of topics to represent data better with this model, we have built our experiment setup. A grid search is conducted for parameters the number of topic words and the number of topic clusters to determine the better combination. The experiment setup has been applied to both Welcome Call and PQM datasets.

- Optimal number of topic clusters
  We chose the number of topic clusters matching the LDA experiments explained in subsection 5.1.1 to compare the results of the two methods. We first started with a 50-150 range of topics with step size 25 and the addition of 250 as an edge case. Moreover, the step size was later reduced to 5 to narrow down the search space based on the observation with a step size of 25.

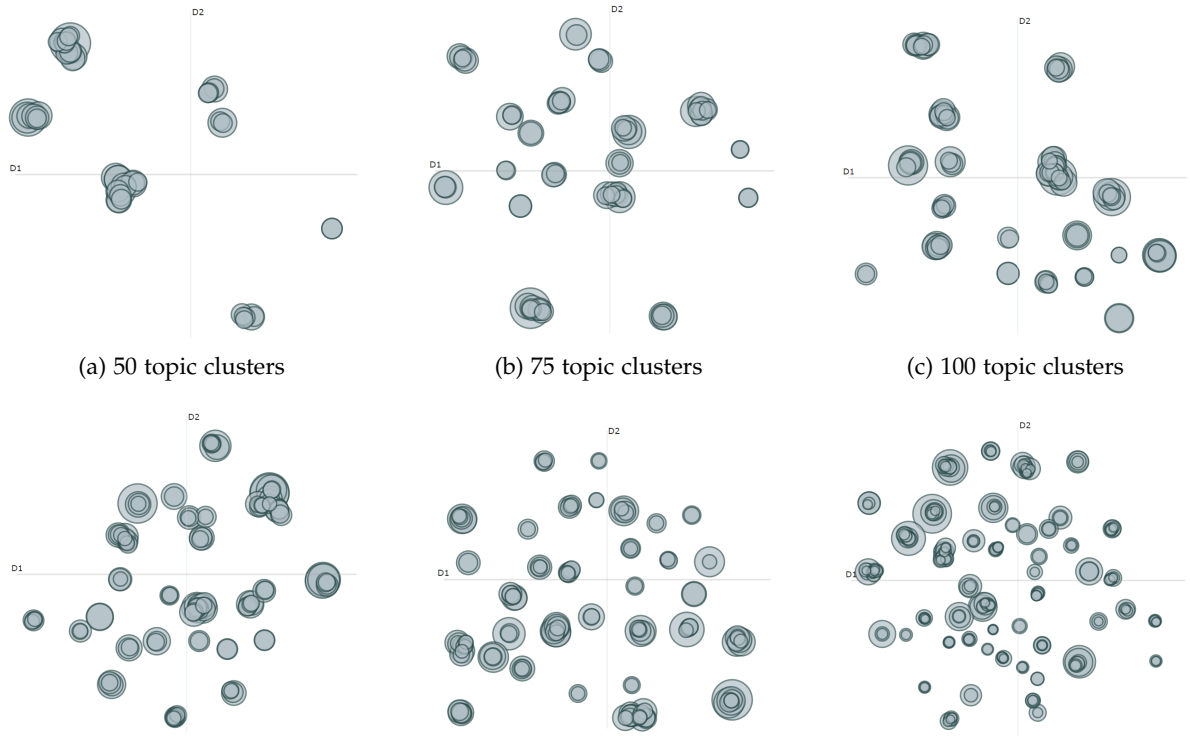
- Number of topic words
  The number of topic words is an important parameter to define what makes a topic. It specifies how many words per topic are used to define a topic. We experimented

with two values for the *top_n_words* parameter, 10 and 20. We wanted to observe the difference it creates in topic representations on each model.

- Interpretation of results
  The decision on the better resulting topic model was concluded by inspecting the intertopic distance maps through a built-in visualization function in the BERTopic library that is implemented through Plotly [91], as well as reviewing topic words to determine the quality of the topic. The *Umass* coherence metric has been reported as the standardized metric.

Welcome Call Dataset

The intertopic distance maps are used to get an overview of the models with different parameter settings. First, we used the top-20-words setting to compare the intertopic distance maps of the different number of topic clusters. Figure 5.9 shows the intertopic distance maps with the top-20-words setting for the different number of topic clusters. It is observable that the topic clusters are not as widely spread as LDA intertopic distance maps, shown previously on Figure 5.1.



(a) 50 topic clusters (b) 75 topic clusters (c) 100 topic clusters

(d) 125 topic clusters (e) 150 topic clusters (f) 250 topic clusters

Figure 5.9.: Comparison of different numbers of topic clusters with BERTopic for Welcome Call by the intertopic distance maps

For 50 topic clusters, we could count six areas where the topic clusters overlap with each other. A deep inspection of the intertopic distance maps showed that the topic clusters are not overlapping, but due to the embedding-based representation, some topics are closer to each other in the latent space. As we zoom into the plots, this can be observed more clearly, as shown in Figure 5.10. It can also be thought that the dataset can be represented in these six areas; however, that would result in very general topics, which does not help to find production issues.

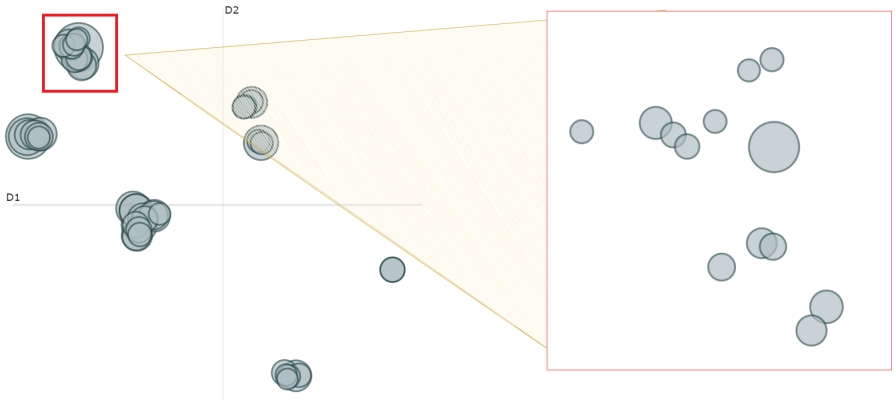


Figure 5.10.: Intertopic distance map for Welcome Call with 50 topic clusters, zoomed

Next, we kept the number of topic clusters steady and compared the number of topic words interpreted through intertopic distance maps. Figure 5.11 shows the intertopic distance map comparison for top-10-words and top-20-words with 75 topic clusters. The top-10-words setting intertopic distance maps were not observed to be too different from the top-20-words setting. Intertopic distance map inspection with different parameters did not yield a clear interpretation of topic models, unlike LDA models; hence we continued our analysis through topic words.

Since the intertopic distance maps did not yield a concrete difference between top-10-words and top-20-words settings, we used topic words to interpret better how this parameter affects the topic quality. We observed that the models with the top-20-words setting had more information stored but also contained more noise. Some comparison examples for the 85 topic clusters model are given on Table 5.4. Some topics are identical to each other, for example, the topic with the keyword '*electronics*', whereas some topics only share a few words in common, like the one with the keyword '*brake*'. We used the top-20-words models for our deep-dive analysis of topic words; however, it is hard to define top-20-words topics as better than top-10-words.

To get a comparative analysis, the same keywords used for our LDA experiments '*brake*', '*seat*', '*mirror*' are used. Table A.5 shows the list of topic words for the given keywords. As the

(a) Top 10 words (b) Top 20 words

Figure 5.11.: Comparison of the intertopic distance map for Welcome Call, top 10 vs. 20 words

number of topic clusters increases, the number of topics occurring with the selected keywords also increases. For example, from the 100 topic clusters model onward, models display more than one topic relevant to that keyword. For the keyword *'seat'*, one topic can be related to seatbelt buckle and comfort, whereas another can be about the leather covering of the seat.

Number of topic words	Keyword	Topic words	Topic ID
10	electronics	electronics, turn, open, door, shut, idle, stay, radio, lock, ignition	1
	brake	brake, collision, launch, pedal, warn, frontal, stop, control, auto, hold	19
20	electronics	electronics, turn, open, door, shut, idle, stay, radio, lock, ignition, vehicle, exit, engine, light, remain, car, setting, set, power, button	1
	brake	brake, noise, squeak, collision, dust, break, stop, squeal, period, mile, warn, normal, pad, rotor, noisy, frontal, hear, sound, apply, speed	4

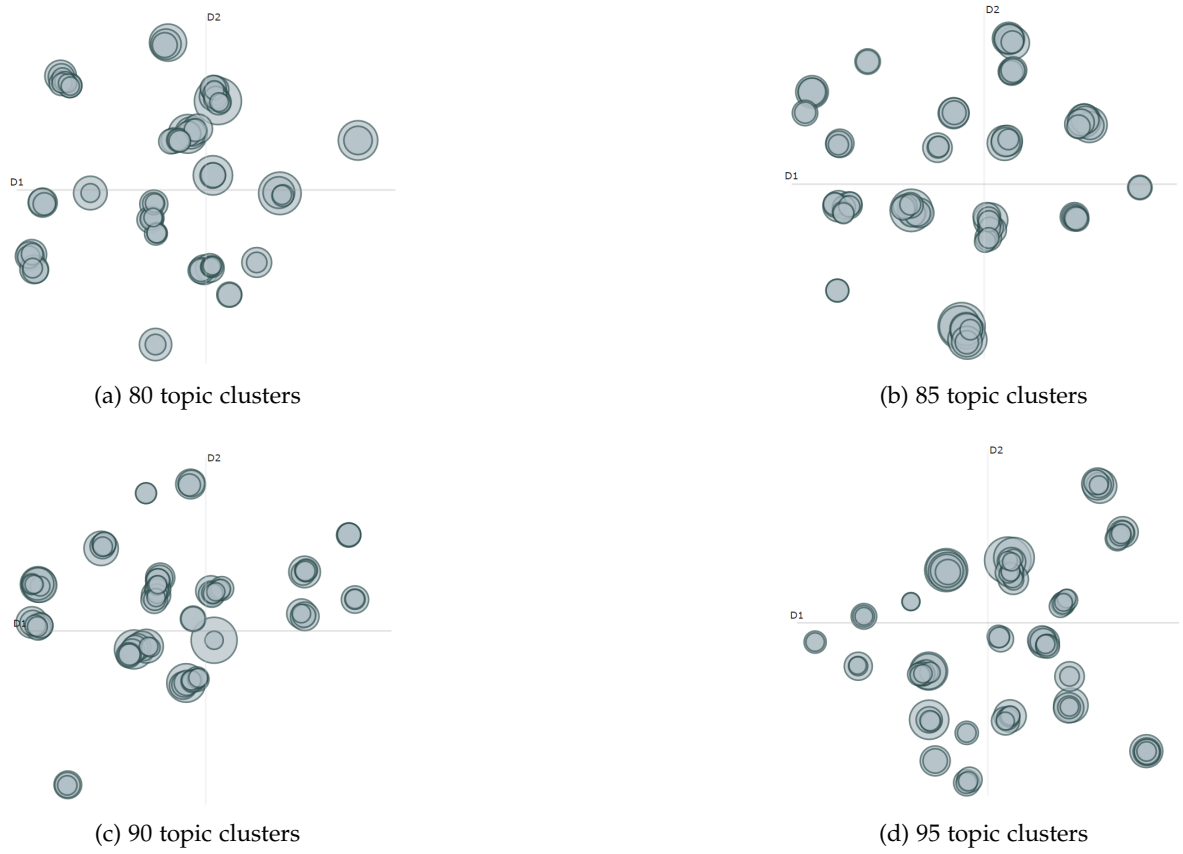
Table 5.4.: Comparison of top-10-words vs. top-20-words setup for 85 topic clusters BERTopic Welcome Call model

Unlike the LDA models, topics continue to give new information on the data with the increased number of topic clusters. For example in 100 topic clusters model the keyword *'spotify'* occurs in Topic 3 with topic words *'app'*, *'assist'*, *'connect'*, *'spotify'*, *'bmw'*, *'account'*, *'log'*, *'alexa'*, *'update'*, *'download'*, while in 150 topic clusters model, the same topic is Topic 1 with similar topic words *'app'*, *'assist'*, *'connect'*, *'bmw'*, *'alexa'*, *'account'*, *'log'*, *'update'*, *'download'*. However, the keyword *'spotify'* does not occur in this topic, because 150 topic clusters model generated a new topic for this keyword with topic words *'spotify'*, *'premium'*, *'qr'*, *'account'*, *'app'*, *'log'*, *'update'*, *'music'*, *'play'*, *'login'*, *'code'*, *'song'* as Topic 123. This shows that in order to get more specific topics, increasing the number of topic clusters is useful with the BERTopic model.

Although we can get more information by increasing the number of topics, when the 250-topic-cluster model is inspected, we observe that several topics could have been clustered

into one topic. For example, thirteen topics involve the keyword *'carplay'*, of which only nine of them have the keyword in the first five topic words. Among these Topic 237 with topic words *'carplay'*, *'com'*, *'mobile'*, *'apple'*, *'connection'* seems like a subset to Topic 11 with topic words *'io'*, *'carplay'*, *'iphone'*, *'connection'*, *'apple'*. Another example is with the keyword *'comfort'*, Topic 133 with topic words *'comfort'*, *'access'*, *'standard'*, *'federally'*, *'mandate'* and Topic 148 with topic words *'comfort'*, *'access'*, *'standard'*, *'feature'*, *'package'* seem like they can be merged into one topic.

Given the above reasonings on the topic words, even though we can continue to get more information by increasing the number of topic clusters, the models with topic cluster numbers 125, 150, and 250 generated some topic clusters that could be part of bigger clusters. Unlike the LDA models with the same settings, meaningless topics are not observed, nor are topic word repetitions in the medium size topic clusters. Overall, generated topics have been observed to be of good quality.



(a) 80 topic clusters

(b) 85 topic clusters

(c) 90 topic clusters

(d) 95 topic clusters

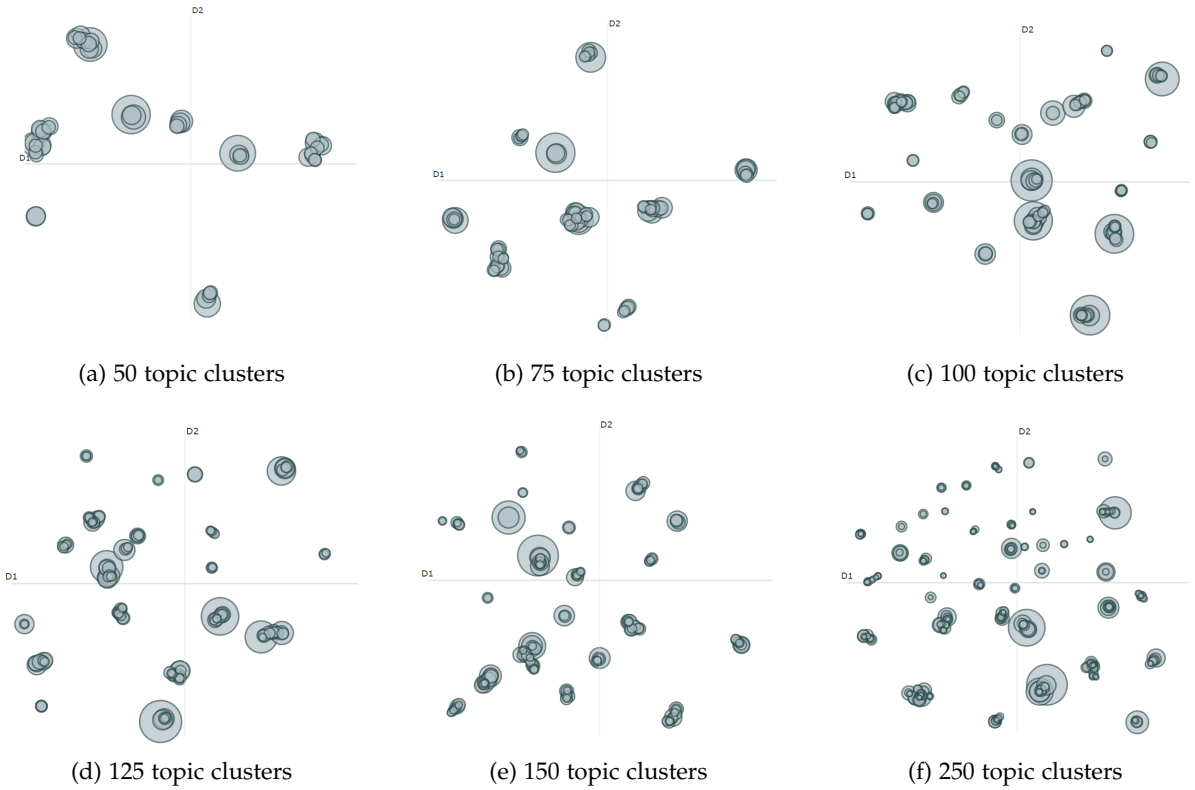
Figure 5.12.: Comparison of different numbers of topic clusters with BERTopic for Welcome Call by the intertopic distance maps, range 80-95

We observed that up to 125 clusters are a suitable representation of the given dataset with the BERTopic model on the experimented range of the number of topic clusters. To

better observe the difference with the LDA model, we applied a deep-dive into the topic cluster range between 75-100 with step size 5. The resulting intertopic distance maps are displayed in Figure 5.12, which do not look that different than each other, similar to the LDA representation of the same topic cluster range; however, as stated before, visibly not as widely spread as the nature of the BERTopic model. We have not observed 'bad' topics in any of the models at the specified range to compare topic words with.

Table A.4 shows the topic words for the keywords '*brake*', '*seat*' and '*mirror*'. We observed multiple topics that contained the specified keywords for each topic model. As an example, in the topics of the 85 topic clusters model, six topics have occurred with the keyword '*seat*', all specifying a different issue related to seats, such as seat heating (Topic 46), the leather cover of the seat (Topic 48), seat position adjustment (Topic 62), seatbelt buckling (Topic 77), etc. For the 80 topic clusters model, the keyword '*seat*' occurs in three different topics, while for the 90 and 95 topic clusters models, it occurs in four topics. The topics generated through the four topic models are all similar to each other, in good quality, and also meaningful.

PQM Dataset

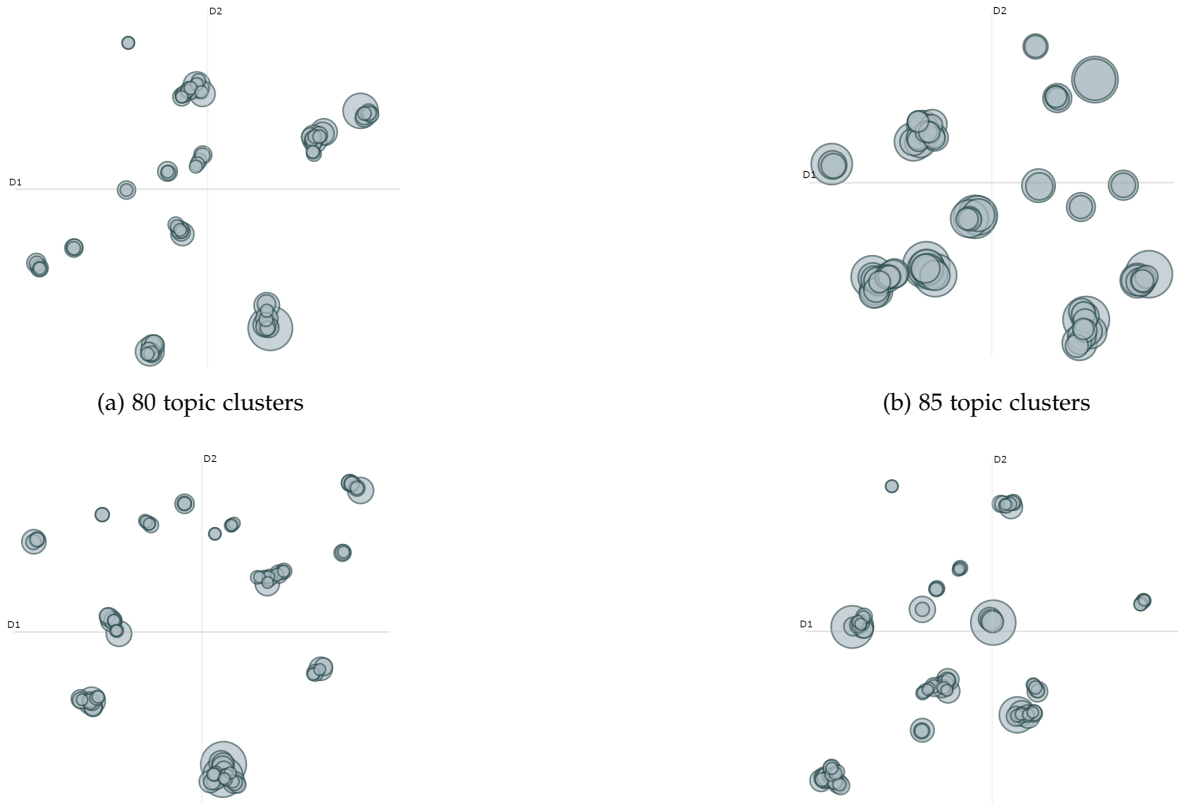


(a) 50 topic clusters (b) 75 topic clusters (c) 100 topic clusters

(d) 125 topic clusters (e) 150 topic clusters (f) 250 topic clusters

Figure 5.13.: Comparison of different numbers of topic clusters with BERTopic for PQM by the intertopic distance maps

Figure 5.13 shows the intertopic distance maps of range 50-150, also the edge case 250 topic clusters, for top-20-words setting generated from PQM dataset. The observations from the Welcome Call dataset experiments also apply to the PQM dataset. 50 topic clusters (Figure 5.13a) show 8 areas of clusters, while as the topic number increases up to 250 (Figure 5.13f) we observe more than 30 areas with much smaller clusters. Intertopic distance maps of 80-95 topic clusters range are shown in Figure 5.14.



(a) 80 topic clusters

(b) 85 topic clusters

(c) 90 topic clusters

(d) 95 topic clusters

Figure 5.14.: Comparison of different numbers of topic clusters with BERTopic for PQM by the intertopic distance maps, range 80-95

Table A.6 for range 50-250 and Table A.7 for range 80-85 shows the topic words for keywords '*brake*', '*seat*' and '*mirror*'. PQM models generated more topics with the selected keywords compared to Welcome Call models. Multiple topics for the given keywords are observed even with lower topic cluster numbers, such as 50. The retrieved topic words for the keyword '*seat*' are about 'seat belt comfort', 'leather cushion of the seat', 'squeaking armrests of the seats', or 'seat heating', which is similar to Welcome Call. As the topic cluster number increase, at 90 topic clusters, it is observable that the keyword '*mirror*' is not just observed in one topic relevant to the side mirrors, but the keyword occurs in topics related to the mirror inside the sun visor. Although from 150 topic clusters onward, we observed that 'the exterior

mirror folding' topic splits into two topics. We also observed the keyword '*brake*' in multiple topics from the 80 topic clusters model. The topics generated with this keyword were hard to interpret without expert knowledge. From 125 topic clusters onward, it was possible to cluster multiple topics together as more subtopics started to appear. Another observation is that due to the structure of this dataset, some of the topics seem meaningless as they repeat some phrases, such as '*Please see the attachment*' or '*Please contact xxx*'. An expert deduction is necessary for these topics to determine whether they are meaningful or not.

Metric Evaluation

Apart from the intertopic distance maps and the interpretation of topic words, we have calculated the *Umass* score for each experiment. The performance table for Welcome Call and PQM is shown in Table 5.5, with the different number of clusters and the number of topic words parameters.

		Welcome Call model		PQM model	
Number of topic words	**Number of topic clusters**	**Minimum number of documents per cluster**	**Umass**	**Minimum number of documents per cluster**	**Umass**
10	50	1645	-2.87	1069	-2.87
	75	1008	-2.92	667	**-2.82**
	80	1016	**-2.78**	612	-2.95
	85	939	-2.86	603	-2.87
	90	807	-2.83	588	-2.96
	95	808	-2.89	512	-2.99
	100	720	-2.95	543	-3.07
	125	550	-2.97	347	-3.09
	150	464	-3.02	305	-3.22
	250	221	-3.15	126	-3.56
20	50	1638	**-3.07**	1030	**-3.06**
	75	1063	-3.12	687	-3.2
	80	947	-3.23	646	-3.19
	85	970	**-3.18**	647	-3.35
	90	890	-3.20	607	-3.36
	95	789	-3.27	544	**-3.28**
	100	723	-3.33	494	-3.44
	125	552	-3.44	342	-3.59
	150	468	-3.58	286	-3.75
	250	217	-3.79	118	-4.52

Table 5.5.: BERTopic model experiment results. For each cluster number, the minimum number of document clusters, which indicates the size of the generated clusters, is reported. As the number of clusters increases, the minimum number of documents in one cluster decreases. *Umass* is reported as the coherence score. The model is considered more coherent as the *Umass* score gets closer to zero.

The *Umass* score is observed to be inversely proportional to the number of topic clusters, similar to the LDA results shown in Table 5.2, with a few exceptions. For the top-10-words setting of Welcome Call models in Table 5.5, the mid-clusters 80 to 95 have a lower *Umass* score than the smaller number of cluster 75. For the top-20-words setting, although the lowest *Umass* score is achieved with 50 topic clusters for both datasets, the 85 topic clusters model for Welcome Call and the 95 topic clusters model for PQM have been observed to break the linear decrease in the score table.

Besides the *Umass* metric, we have also reported the minimum number of documents per cluster to compare cluster sizes for different cluster numbers. It is observable that as the number of topic clusters increases, the document number in the smallest cluster decreases.

5.1.3. Comparison

In the above sections, we have explained our experiment setups and demonstrated our results for the Welcome Call and PQM datasets by applying LDA and BERTopic methods. In this section, to conclude our topic modeling experiments, we will discuss the differences we observed between the two methods and introduce the reasoning for our next experiment. As discussed earlier, we compare the models in three aspects: evaluation metrics, intertopic distance maps, and topic words.

Evaluation Metrics

Figure 5.15 and Figure 5.16 displays the model performance comparison for both models based on the *Umass* score. By default, the Coherence Model implementation of gensim[89] uses the top 20 words per topic to calculate the coherence metric. Users can specify this parameter as a hyperparameter during BERTopic training; however, the LDA generates the top 30 words for each topic by default. Our experiments explored the top 10 words and top 20 words settings for BERTopic. In order to compare the two models fairly, we calculated the *Umass* score with the top 10 words for the LDA model as well. We observed that this parameter significantly affects the generated coherence scores. An increase in the topic words affects the scores negatively; this is observable in both models but especially significant in the LDA model. Two lines of the LDA model in both figures go parallel to each other until the topic cluster number becomes too high, as in more than 150 clusters. Based on the *Umass* scores, topic coherence of the BERTopic models has resulted better than the LDA models.

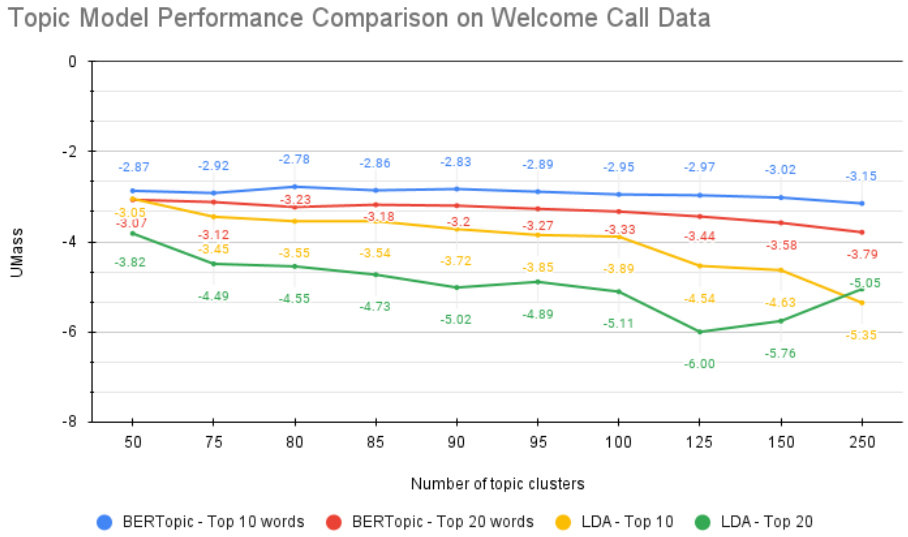


Figure 5.15.: Topic Model Performance on Welcome Call dataset. BERTopic and LDA are compared with the top-10-words and the top-20-words settings. *Umass* scores for each number of topic clusters are reported. The closer the *Umass* score is to zero, the better the model is.

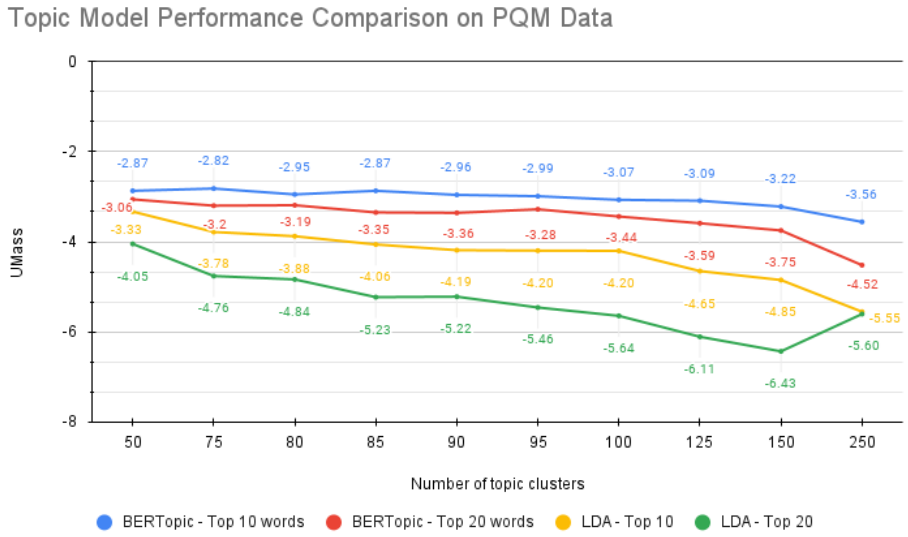


Figure 5.16.: Topic Model Performance on PQM dataset. BERTopic and LDA are compared with the top-10-words and the top-20-words settings. *Umass* scores for each number of topic clusters are reported. The closer the *Umass* score is to zero, the better the model is.

Topic Words

One noticeable difference between BERTopic and LDA is that BERTopic creates a class labeled with '-1' and clusters the documents that are considered outliers under this label; that is why the topic words do not contain many outlier words like the LDA models. 'Topic -1' is not displayed as part of the intertopic distance maps. This feature is beneficial if the aim is to discover more specific topics rather than general clustering. However, one disadvantage is that the number of documents clustered as outliers is much higher than the desired amount, so much so that Topic -1 is the most frequent topic among the generated topics. The rest of the topic labels are ascendingly sorted by the frequency of documents assigned to the topics, as in, Topic 0 would have the most number of documents, and Topics 1, 2, 3, and others follow. LDA does not differentiate outlier documents, and because of this, the generated topics can contain many outlier words, which reduces overall topic quality.

When we checked topics with relevant keywords, BERTopic resulted in more specified topic clusters on the production-related keywords compared to LDA. Statistical method LDA clustered the topics based on the probability of the words occurring in the documents. The generated clusters by this method would be considered satisfactory if the aim is to get a general clustering of the documents; however, since it is a context-independent method, the generated topics were not as informative as BERTopic when it comes to production-specific topics. Due to its hierarchical clustering approach and also knowledge of the context, topics generated by BERTopic are more meaningful than LDA. These differences between the approaches cause the LDA topics to be not meaningful in our case.

Intertopic Distance Maps and Optimal Number of Topic Clusters

Finding the optimal number of topic clusters was a challenge for both models. The BERTopic model can auto-generate the optimal cluster number; however, during our experiments, we observed that the auto-generated cluster number depends on the minimum documents per topic cluster parameter. The generated topics turned out to be too many due to the value of this parameter, and some of the observed topics were repetitive. Hence this feature proved not helpful for our task, and we had to search for the optimal number of topic clusters. LDA model requires this search since the number of topic clusters parameter is required for the algorithm to work.

Intertopic distance maps are suitable for analyzing the topic structure of the model as an overview. The larger the circle in the plot, the more prevalent that topic is. The intertopic distance maps showed that the LDA clusters are more spread than the BERTopic clusters, which shows topic diversity. LDA clusters can be overlapping even though they seem more spread, whereas BERTopic clusters seem as if they are more overlapping, whereas they are not.

The optimal cluster number for the given datasets is observed to be between 75 and 100 for both methods. We observed that LDA does not provide a good topic representation for the high number of clusters for the given datasets, while BERTopic continues to provide a fitting representation with the higher number of clusters. The higher cluster models can be seen to

be repetitive but still meaningful with BERTopic, while with LDA, it is not only repetitive but also not meaningful given the nature of our datasets. For lower clusters, the quality of LDA topics is observed to be close to BERTopic, although the clustering did not give the production-related clusters we aimed for.

Decision

Given the discussion above, we decided to move forward with BERTopic. However, it was still unclear how many topics would better represent the data; hence we decided to compare 80, 85, 90, and 95 topic cluster models with both top-10-words and top-20-words settings for each dataset in the next phase of our research.

5.2. Topic Matching

For our topic matching experiments, we used two similarity measures: Cosine Similarity and Soft Cosine Similarity. For Cosine Similarity implementation, sklearn [92] library is used. For Soft Cosine Similarity implementation [93] we used gensim[89] library using Glove model [17] as the relation matrix, given that several studies have reported that Glove embeddings have performed better than Word2vec embeddings [94] [95] [96]. We used the topic words and topic embeddings as inputs to our topic matching/text similarity pipeline. As stated earlier, it was unclear which combination of hyperparameters for the topic modeling was best; for this reason, we have calculated similarities between the 80, 85, 90, and 95 topic cluster models for top-10-words and top-20-words settings.

We chose to separate the calculation between the top-10-words models and the top-20-words models since the scoring would not be fair if the number of topic words is not the same, as seen on Figure 5.15 and Figure 5.16. We have compared four topic clusters with two different topic word numbers for two datasets, which resulted in $4 \cdot 4 \cdot 2 = 32$ different combinations of model pairs.

5.2.1. Experiment Setup

The steps for the experiment setup are as follows:

1. Load models of Welcome Call and PQM with the selected cluster numbers.
2. Create labels for each topic belonging to models.
3. Apply the selected similarity metric to calculate the distance matrix.
4. Visualize the distance matrix as a heatmap.
5. Convert the distance matrix to a data table, appending similarity scores with the data labels.
6. Analyze and filter the data table to get newly discovered topics from the Welcome Call model.

Cosine Similarity

In section 4.2, we discussed that our application of Cosine Similarity would adapt based on the selected topic model method. Since BERTopic was used as the selected topic model, we used the topic embeddings generated by the model as input to Cosine Similarity instead of the topic words to calculate a distance matrix for each model pair. sklearn [92] implementation was used to calculate 2D-matrix level similarity calculations. As explained in Equation 2.1, this function takes two matrices as input; in this case, one of the matrices is the topic embeddings of the selected Welcome Call model, and the other one is from the PQM model. The values of the resulting distance matrix range between 0 and 1 as the topic embeddings are non-negative.

Soft Cosine Similarity

For Soft Cosine Similarity, we used gensim [89] implementation. As explained in subsection 2.4.2, and stated in Equation 2.2, this metric takes three matrices as input, two of them coming from the data and the third one is the contextual word embedding. We used the GloVe[17] model '*glove-wiki-gigaword-50*' as the word embedding matrix. The implementation creates a dictionary with the words from both models. It also requires two input queries to be provided and converts these queries to TF-IDF representation by using the provided dictionary, which contains the topic words that occur in both datasets.

Similarity Table

The output of the similarity calculations between the two models results in a distance matrix. This distance matrix is converted into a table for interpreting the generated scores easily. The table has 4 columns: 'Similarity Score', 'Rounded Similarity Score', 'Welcome Call topic name', and 'PQM topic name'. Topic names are generated by concatenating topic IDs and topic words together in the format of '*topicID_word1_word2...wordn*'. The rounded similarity score is the similarity score rounded up to 3 decimals; since the generated similarity score is by default with six decimals, and some of the values are observed to be very close to each other, we used the 'Rounded Similarity Score' to determine the quality of topic matches. The resulting table length is $M * N$, M being the number of topic clusters of the Welcome Call model, and N being the number of topic clusters of the PQM model. For example, if the Welcome Call model has 80 topic clusters and the PQM model has 90 topic clusters, the table length would be 7200.

After this conversion to the table, we filtered the table to have only the highest values for each Welcome Call topic for the selected model pairs, 'Rounded Similarity Score' is used for this step. This filtered table contains every Welcome Call topic name from the original table; however, depending on the similarity scores, not every PQM topic name is part of the table. The length of the table then would be either M or slightly more than M since the rounded similarity scores can be the same for multiple topic pairs with the same Welcome Call topic name.

5.2.2. Evaluation

We analyzed each possible combination of the previously mentioned topic models with the specified topic cluster numbers to evaluate topic matching experiments. A threshold was set for each similarity metric, and it was considered a binary classification problem. If the similarity score between two topics is higher than the defined threshold, this topic pair is considered a *Match*, and if it is lower, it is a *Non-match*. After inspection of similarity tables, it was observed that the thresholds for Cosine Similarity and Soft Cosine Similarity could not be the same, as the Cosine Similarity between topics can reach up to 0.99; however, the Soft-Cosine Similarity does not surpass 0.75. For the lowest value, Cosine Similarity does not go below 0.6, whereas Soft Cosine Similarity can go down to 0.12.

It was decided that a suitable threshold for Cosine Similarity could be 0.85 and for Soft Cosine Similarity 0.4. The similarity tables were labeled by hand to test this hypothesis, and a confusion matrix was created per model pair. Precision, Recall, and F1 score are reported as evaluation metrics for this classification task based on the hand-labeled topic pairs. The evaluation results for the similarity matches between topic models for both Cosine and Soft Cosine Similarity measures are reported in Table 5.6.

Cosine Similarity resulted in a 1.00 precision score for three combinations of top-10-word setups. The top-20-words setup achieved the highest precision score of 0.897, with the Welcome Call 80 clusters model and PQM 90 clusters model pair. For recall, a score of 1.00 is achieved through the 85-85 pair with the top-20-words setup, followed by the 90-95 pair with a score of 0.972. Three combinations obtain the third highest score of 0.969 for top-20-words: 80-80, 80-95, and 85-95 pairs. The top-10-words models were observed to have higher precision and lower recall scores compared to the top-20-words models.

The best F1 score for top-10-words is obtained between Welcome Call 90 clusters model and PQM 95 clusters model, with a score of 0.885. For the top-20-words setup best F1 score was reported as 0.921, which is reported by two Welcome Call - PQM model pairs: 80-90, which also has the best precision for this setup, and 90-95. These are followed by an F1 score of 0.899 obtained with the 85-95 model pair. Five combinations of the top-20-words setup surpassed the highest F1 score for the top-10-words setup.

The best F1 score for Soft Cosine Similarity for the top-10-words setup is obtained between Welcome Call 80 clusters model and PQM 95 clusters model, with 0.736; the same model pair has also reported the best precision and best recall. For the top-20-words setup, the best precision was obtained by the 90-90 pair, whereas the 85-95 pair obtained the best F1 score and best recall. It was observed that Soft Cosine Similarity did not result in higher evaluation scores on topic matching than Cosine Similarity, which was expected as the embeddings used for Cosine Similarity are directly generated through the topic model's sentence-level embeddings, and as for Soft Cosine Similarity, a fixed-length word embedding (GloVe) is used. This difference between the embeddings could significantly affect the performance since the fixed-length word embeddings do not provide any contextual information for the production domain, whereas the contextual embeddings were fine-tuned for the specific domain. This explains why measuring the similarity with contextual embeddings yields better results than using fixed-length word embeddings.

Number of topic words	Number of Welcome Call clusters	Number of PQM clusters	Cosine similarity			Soft Cosine Similarity		
			Precision	Recall	F1 score	Precision	Recall	F1 score
10	80	80	0.957	0.759	0.846	0.568	0.727	0.638
		85	**1.000**	0.719	0.836	0.583	0.714	0.642
		90	0.875	0.724	0.792	0.595	0.682	0.635
		95	0.885	0.742	0.807	**0.659**	**0.833**	**0.736**
	85	80	0.963	0.813	0.881	0.600	0.696	0.644
		85	0.852	0.719	0.780	0.592	0.769	0.669
		90	0.929	0.765	0.839	0.622	0.680	0.650
		95	0.935	0.829	0.879	0.653	0.680	0.666
	90	80	**1.000**	0.788	0.881	0.548	0.792	0.647
		85	0.917	0.759	0.830	0.556	0.741	0.635
		90	**1.000**	0.727	0.842	0.548	0.826	0.659
		95	0.844	0.931	**0.885**	0.625	0.652	0.638
	95	80	0.844	0.750	0.794	0.595	0.810	0.686
		85	0.963	0.703	0.813	0.622	0.739	0.676
		90	0.871	0.730	0.794	0.568	0.826	0.673
		95	0.750	0.727	0.738	0.638	0.739	0.685
20	80	80	0.795	**0.969**	0.873	0.636	0.696	0.665
		85	0.800	0.941	0.865	0.659	0.652	0.656
		90	**0.897**	0.946	**0.921**	0.674	0.667	0.671
		95	0.816	**0.969**	0.886	0.674	0.789	0.727
	85	80	0.644	0.935	0.763	0.667	0.789	0.723
		85	0.767	**1.000**	0.868	0.640	0.900	0.748
		90	0.778	0.946	0.854	0.681	0.882	0.769
		95	0.838	**0.969**	**0.899**	0.660	**0.947**	**0.778**
	90	80	0.674	0.935	0.784	0.702	0.667	0.684
		85	0.860	0.902	0.881	0.660	0.850	0.743
		90	0.875	0.897	0.886	**0.721**	0.667	0.693
		95	0.875	**0.972**	**0.921**	0.706	0.750	0.727
	95	80	0.744	0.889	0.810	0.708	0.700	0.704
		85	0.756	0.912	0.827	0.642	0.950	0.766
		90	0.783	0.973	0.867	0.711	0.813	0.758
		95	0.825	0.892	0.857	0.692	0.842	0.760

Table 5.6.: Evaluation results of similarity experiments, bold and highlighted values show the highest obtained scores and are discussed under subsection 5.2.2. All scores range between 0 and 1.

5.3. Results

The 85-95 pair with the top-20-words setting was chosen to be used in the web application. This decision was made by taking both Cosine Similarity, and Soft-Cosine Similarity results into account. This pair has the highest F1 score with Soft Cosine Similarity and the second-highest F1 score with Cosine Similarity. Heatmap and Word Cloud visualizations are also used to determine the selected model pair's topic quality. Generated visualizations are used in the designed web application explained in chapter 6, and the results are evaluated with a user study presented in chapter 7.

Heatmap

The similarity of topics for the selected model pair is visualized with a heatmap using the distance matrix generated with Cosine Similarity, shown in Figure 5.17. Heatmap provides an overview of the similarities between the topics of the two models. We used a red-blue color palette to contrast the differences better. Red squares represent the less similar topic pairs, and the blue ones show the more similar topics. It is observable through the vertical red lines that the PQM dataset has several different topics than the Welcome Call dataset, which was expected. Through the horizontal red lines, it can be observed that the Welcome Call dataset also has some topics that are more different than most PQM topics; this confirms our assumption that the Welcome Call dataset contains issues that are not observed in the PQM dataset.

Word Clouds

Some of the topic matches with the highest similarity from the Welcome Call model and the PQM model are shown side by side in Figure 5.18. It can be confirmed via word clouds that these topics match, although the frequency of the documents with the generated topics can differ between datasets, as indicated by the topic IDs. For example, Figure 5.18a shows that the topic about 'seatbelt buckle' topic that occurs much more frequently in PQM dataset as it is the 6th most frequent topic (Topic 5), whereas for Welcome Call it is Topic 77. For Welcome Call dataset, the topic about 'squeaking noisy brakes' occur more frequently than PQM dataset as it is Topic 4 (Figure 5.18c).

Figure 5.19 displays some of the newly discovered issues from the Welcome Call data. Among 85 topics, 47 of them were resulted to be not-matching with topics from the PQM model. We observed that the newly discovered issues are mostly about the vehicle's software. Topic 0 (Figure 5.19a), Topic 3 (Figure 5.19c), Topic 6 (Figure 5.19d) and Topic 7 (Figure 5.19e) are some examples of software-related topics. However, topics such as Topic 10 (Figure 5.19g), Topic 12 (Figure 5.19i) and Topic 16 (Figure 5.19l) are not software-related and still did not match with any PQM topics.

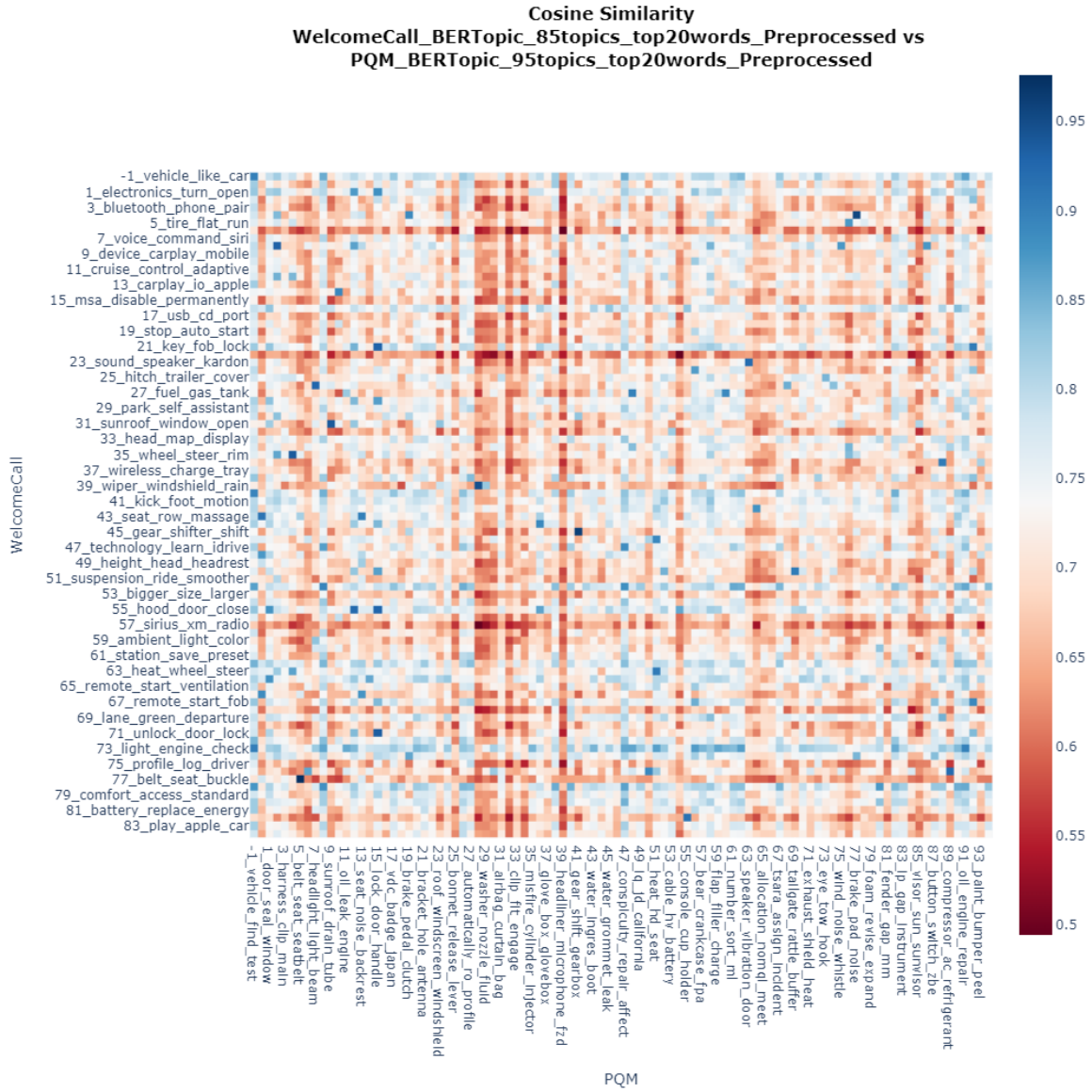
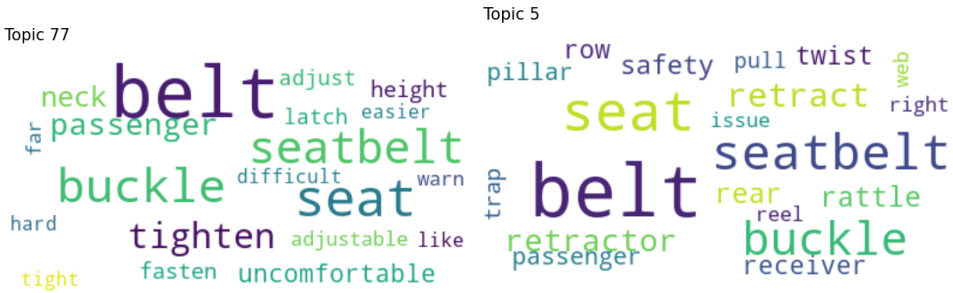
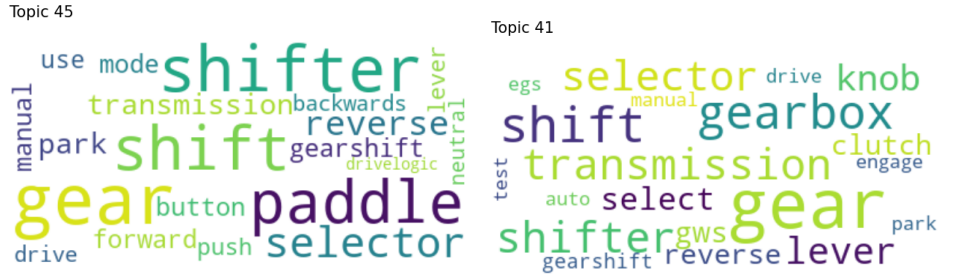


Figure 5.17.: Heatmap comparison of Welcome Call 85 topic cluster model and PQM 95 topic cluster model. Red squares show the dissimilar topics; blue squares show the more similar ones.



(a) Welcome Call Topic 77 and PQM Topic 5, with 0.976 Similarity

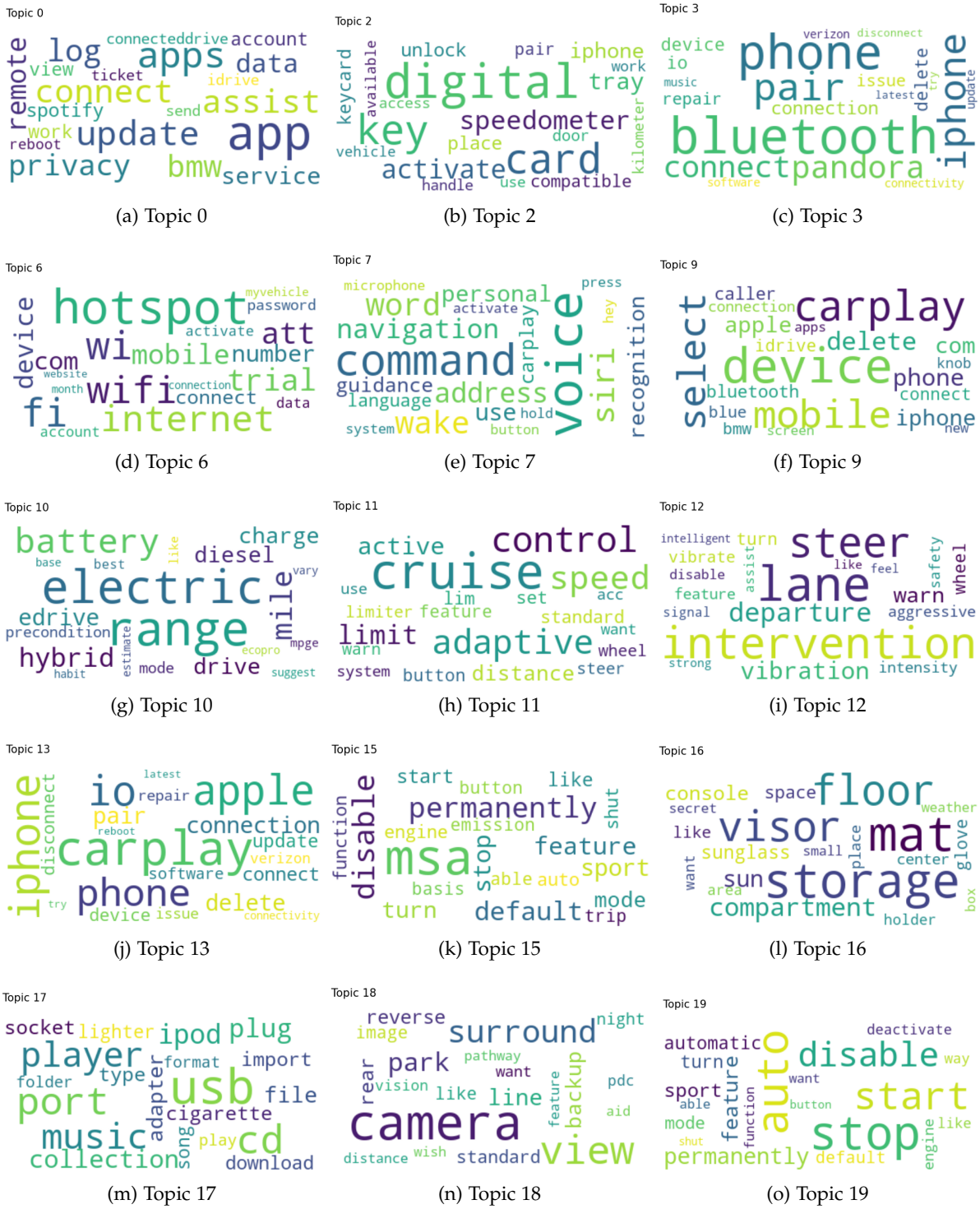


(b) Welcome Call Topic 45 and PQM Topic 41, with 0.962 Similarity

(c) Welcome Call Topic 4 and PQM Topic 77, with 0.956 Similarity

(d) Welcome Call Topic 39 and PQM Topic 28, with 0.948 Similarity

Figure 5.18.: Example topic matches from Welcome Call 85 topic clusters model and PQM 95 topic clusters model.



(a) Topic 0 (b) Topic 2 (c) Topic 3

(d) Topic 6 (e) Topic 7 (f) Topic 9

(g) Topic 10 (h) Topic 11 (i) Topic 12

(j) Topic 13 (k) Topic 15 (l) Topic 16

(m) Topic 17 (n) Topic 18 (o) Topic 19

Figure 5.19.: Newly discovered topics from Welcome Call dataset, with 85 topic clusters model

6. System Architecture

Target users for this project are BMW Group employees with domain knowledge in datasets mentioned in chapter 3. Technical knowledge of the users can vary depending on their background, so the interface needs to be understandable. The final product of this work is a Flask [11] based web application. Following the results of experiments mentioned in chapter 5, the system uses two BERTopic models fine-tuned with Welcome Call and PQM datasets as topic models, and the similarity calculations are conducted with Cosine Similarity applied on the topic embeddings of the models. The application has two available options for the user:

1. Use system documents: Documents that are used during model training to obtain topics.
2. Upload new documents: The system can use the trained models to predict the topics of unseen documents.

This chapter is divided into two sections, explaining the front-end in the first section and the back-end in the second section. System architecture is shown in the Figure 6.1.

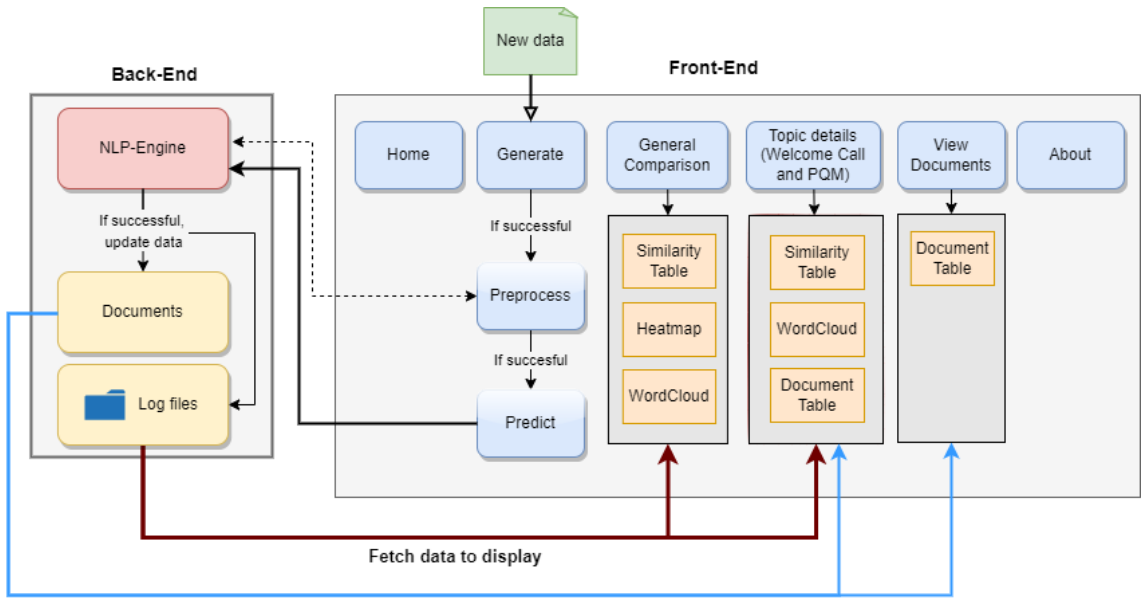


Figure 6.1.: System architecture of the Framework showing the interaction between back-end and front-end. The blue squares are the pages; the orange squares are the features displayed on the pages. Yellow squares are the documents and static system files.

6.1. Front-End

For front-end implementation HTML, CSS, JavaScript, Bootstrap templates [97] and Datatables [98] are used. The user interface has the following tabs: 'Home', 'Generate', 'General Comparison', 'Welcome Call Topic Details', 'PQM Topic Details', 'View Documents', and 'About'. The usage of each tab and also the connector pages are discussed below.

6.1.1. Home

This is the application's start page, which gives information on navigating between the tabs. It has hard-coded HTML text and two buttons. The 'Upload files' button redirects the user to the 'Generate' tab for file upload, and the 'Continue with system data' button redirects to the 'General Comparison' page and displays the features using system data used for model training (explained in chapter 3). Figure 6.2 shows a screenshot of the home page.

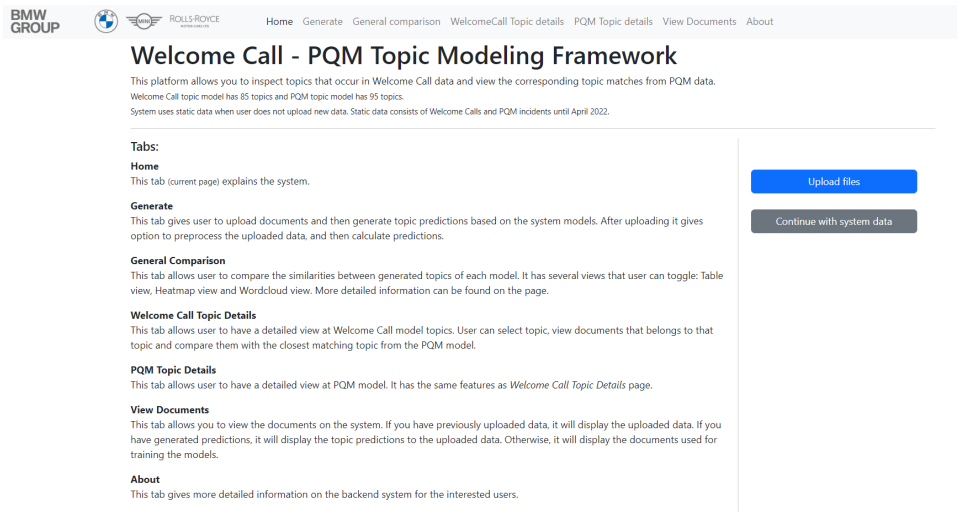


Figure 6.2.: Home page of the Framework

6.1.2. Generate

This tab allows users to upload files from the file system. After submitting the files, it redirects the user to the preprocessing subpage, explained in subsubsection 6.2.1, which is connected to the NLP-Engine in the back-end. A snapshot of the page layout is shown in Figure 6.3.

6.1.3. General Comparison

The general model comparison page provides insights on the similarities and differences between Welcome Call topics and PQM topics. Three sub-tabs exist on this page: 'Table view', 'Heatmap view', and' Word Cloud view'.

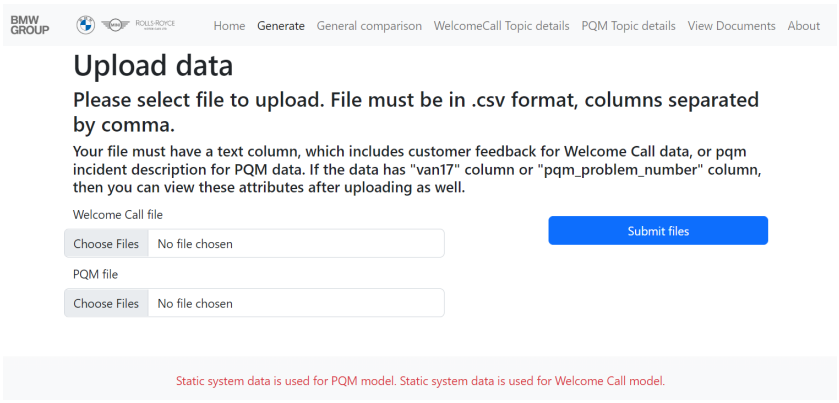


Figure 6.3.: Generate page of the Framework

Table View

Figure 6.4 shows a snapshot from the Table view, which allows the user to observe the similarity between topic matches in detail. It shows two tables through the tabs, **Filtered** and **All**. The table structure for both tables is the same, whereas the entry numbers are different. Table attributes are explained in Table 6.1. Both table views are orderable through numeric values and searchable for topic ids and topic words; for this feature, Datatables plug-in [98] is used. The user can reach the 'Topic details' page by clicking on topic ids.

Column Name	Description
Similarity Score	The cosine similarity score between corresponding topics. (numeric)
WelcomeCall topic id	WelcomeCall model topic ID. (numeric)
WelcomeCall topic words	Topic words for the given topic ID for the WelcomeCall model. (text)
WelcomeCall document frequency	Number of documents that belong to the given topic for WelcomeCall. (numeric)
PQM topic id	PQM model topic ID. (numeric)
PQM topic words	Topic words for the given topic ID for the PQM model. (text)
PQM document frequency	Number of documents that belong to the given topic for PQM. (numeric)
Match Type	Labeled type of match, three categories available: Definite Match, Possible Match, Not Match. (categorical)

Table 6.1.: Similarity table column metadata

- **Filtered** tab allows the user to view the closest PQM matches for each Welcome Call topic. This tab has 102 entries. The table is initially sorted in descending order by the Similarity Score column.

- **All** tab shows the detailed similarity table for every combination of the Welcome Call topics and the PQM topics. This tab has 8075 entries, in ascending order by the Welcome Call Topic id column.

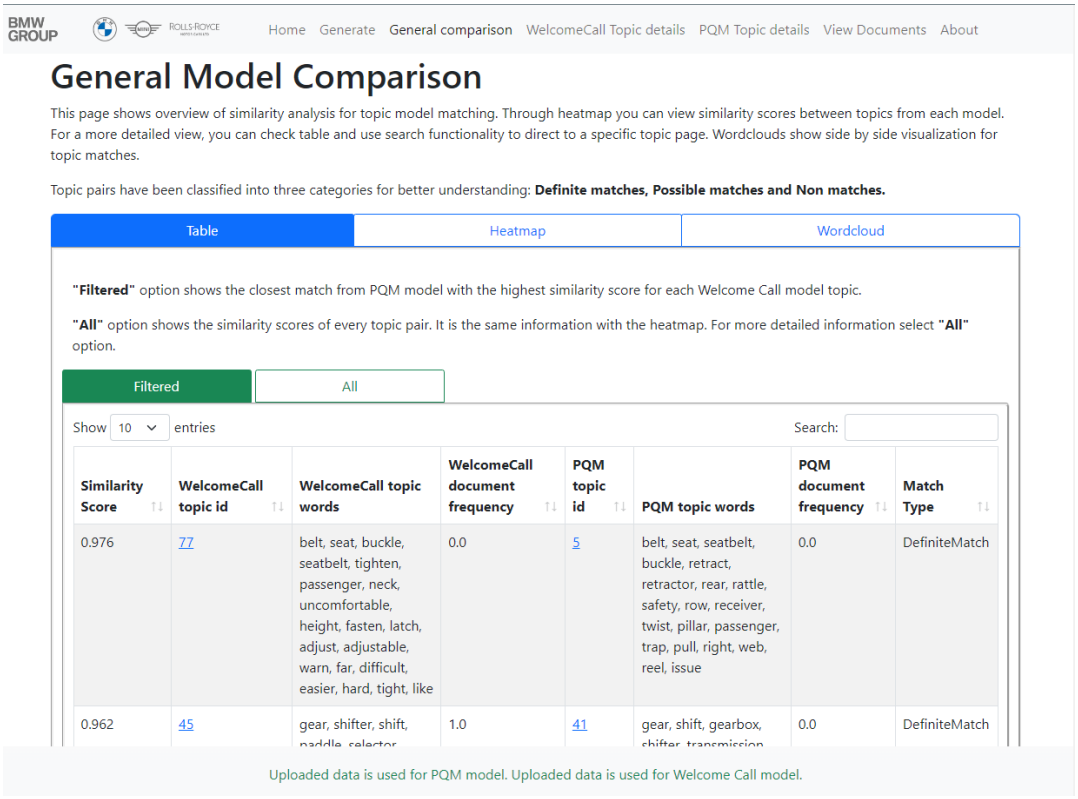


Figure 6.4.: Table view tab of general comparison page

Heatmap View

The Heatmap view shows the same information with the **All** tab from the Table view. Through interactive visualization, it provides the user with a quick overview of the similarities between topics. Red squares show lower similarity, and blue ones show higher similarity. Figure 6.5 shows a snapshot of Heatmap View.

Word Cloud View

The Word Cloud view shows topic matches based on the Filtered similarity table. There are three tabs to navigate inside the page based on Match Type mentioned in subsubsection 6.1.3: Definite-Matches, Possible-Matches, Non-Matches. On the left, it displays the Welcome Call topic Word Cloud, and on the right, it displays the closest matching the PQM topic Word Cloud. Figure 6.6 shows the Word Cloud view.

Figure 6.5.: Heatmap view tab of general comparison page

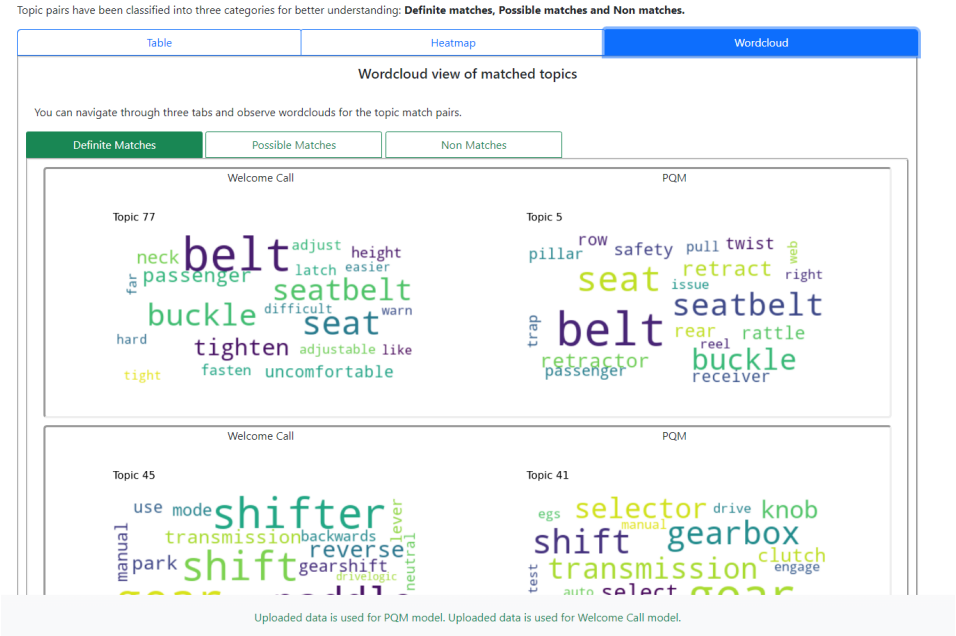


Figure 6.6.: WordCloud view tab of general comparison page

6.1.4. Topic Detail Pages

Topic details can be reached through the navigation bar under the 'Welcome Call Topic details' tab for the Welcome Call topic model and under the 'PQM Topic details' tab for the PQM topic model, as well as through the similarity tables explained in subsubsection 6.1.3 and from the tables on 'View Documents' tab explained in subsection 6.1.5. The layout of the page uses the same template for both models, shown in Figure 6.7. It allows the user to navigate through topics with a dropdown menu. The information shown on this page for the selected topic is listed as follows:

- Topic Word Cloud, which shows the topic words in Word Cloud visualization for selected topic id.
- Number of documents that belong to the selected topic id.
- Similarity table, which displays the similarity scores between the selected topic and all topics from the other topic model. Similarity scores are sorted in descending order. The table is searchable and sortable.
- Topic documents tables, displayed side by side. The first one displays the documents that belong to the selected topic, and the second one with the closest matching topic documents from the other topic model. If provided in the data, it shows van17 numbers for Welcome Call documents and PQM Problem Number for PQM documents.

6.1.5. View Documents

This tab allows the user to view the documents used in the system. Two tabs are displayed, one for Welcome Call documents and one for PQM documents. This page also has a 'Clear Documents' button to reset the system to its original state with static system documents. A page snapshot is shown in Figure 6.8.

6.1.6. About

This tab gives information to interested users on the configurations of the trained models.

6.2. Back-End

Flask [11] API is used as back-end. Flask depends on the Jinja template engine and the Werkzeug WSGI toolkit. [1] When the application starts, the following steps occur:

1. Bertopic pipeline and Data Preprocessing pipelines are initialized.
2. Static log files for similarity tables and system data are initialized and loaded into the system.

[1] https://flask.palletsprojects.com/en/2.2.x/

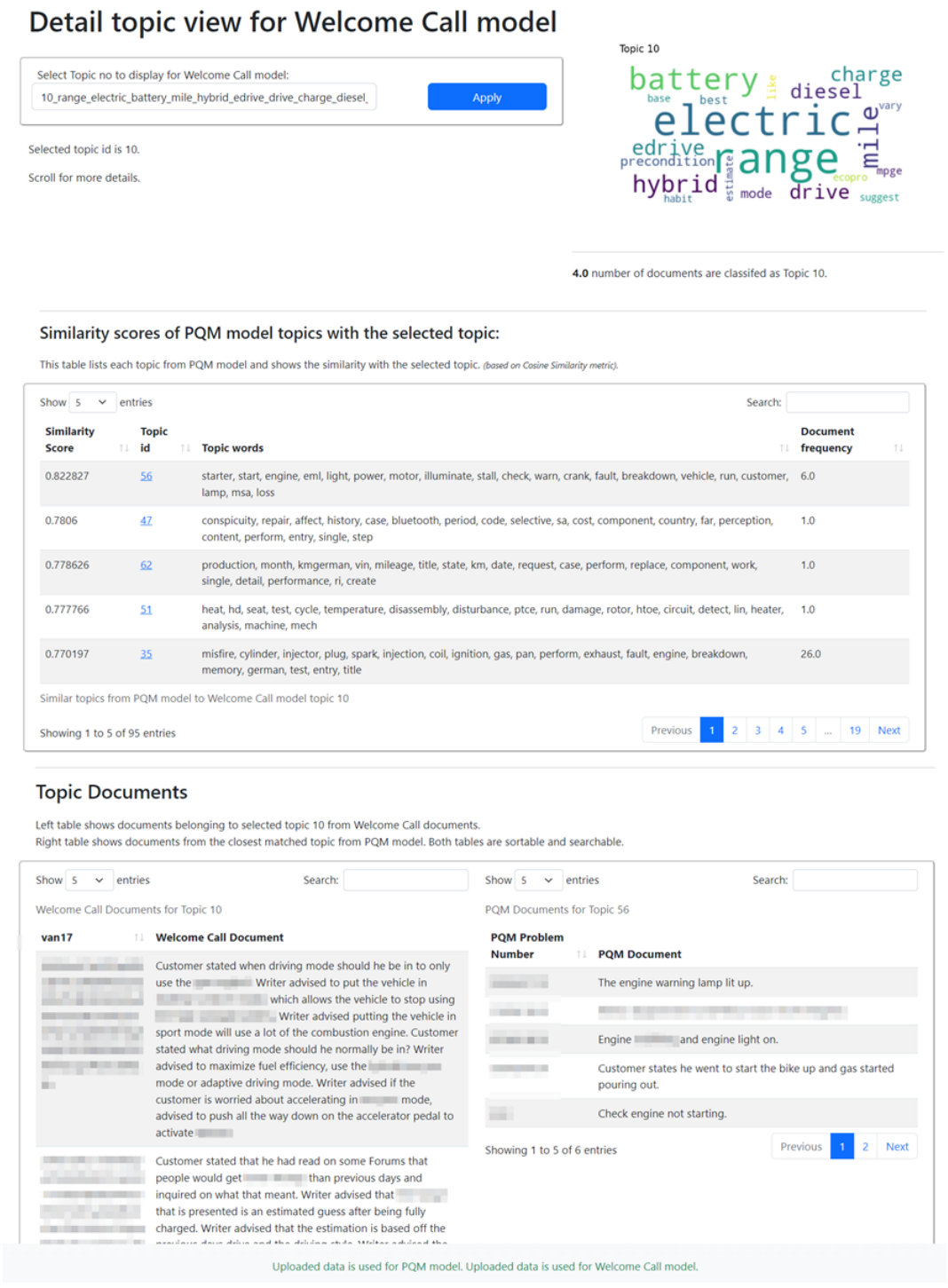


Figure 6.7.: Topic details page

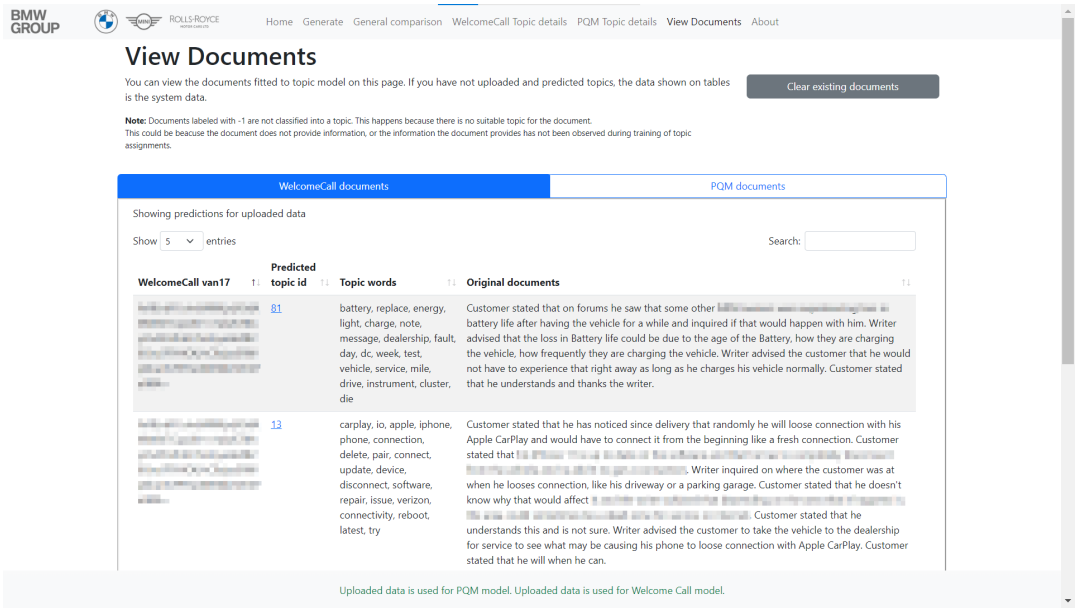


Figure 6.8.: View documents page after prediction

6.2.1. Generating Process

This process starts with the 'Generate' tab, which is reachable from the navigation bar and also through the 'Home' page, shown in Figure 6.3. It allows the user to select files to upload for both Welcome Call and PQM data. Figure 6.9 shows the flow diagram for this tab.

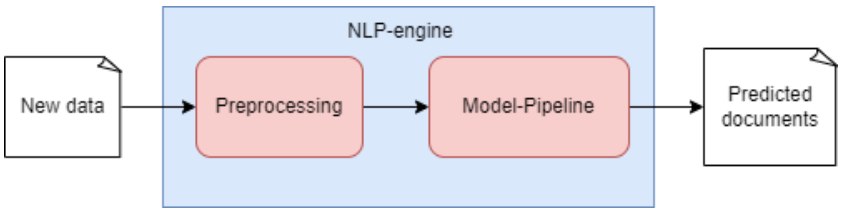


Figure 6.9.: Flow Diagram of generating new predictions

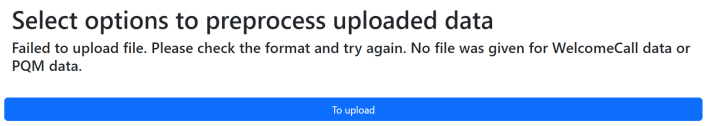


Figure 6.10.: Error message for uploading data

After the 'Submit files' button is clicked, it redirects the user to preprocessing subpage. The back-end process between the two pages occurs as follows:

1. It checks if the files are valid and show an error message if not valid, as shown in Figure 6.10.

2. If files are valid, it uploads the files to the system and displays the preprocessing selection page.

Preprocessing Selection Page

This page is used to configure the parameters necessary for preprocessing the uploaded data. It uses the preprocessing pipeline explained in section 3.3. Two parameters are asked for each uploaded file, (1) text column and (2) stop words. The text column is the column that is to be processed within the NLP-Engine in the back-end, the column options for the dropdown are extracted through the columns of the uploaded data. Stop words are the words that are desired to be removed from the documents. After the necessary parameters are selected, and the 'Set Parameters' button is clicked, it forwards the form input to Preprocessing Pipeline of the NLP-Engine.

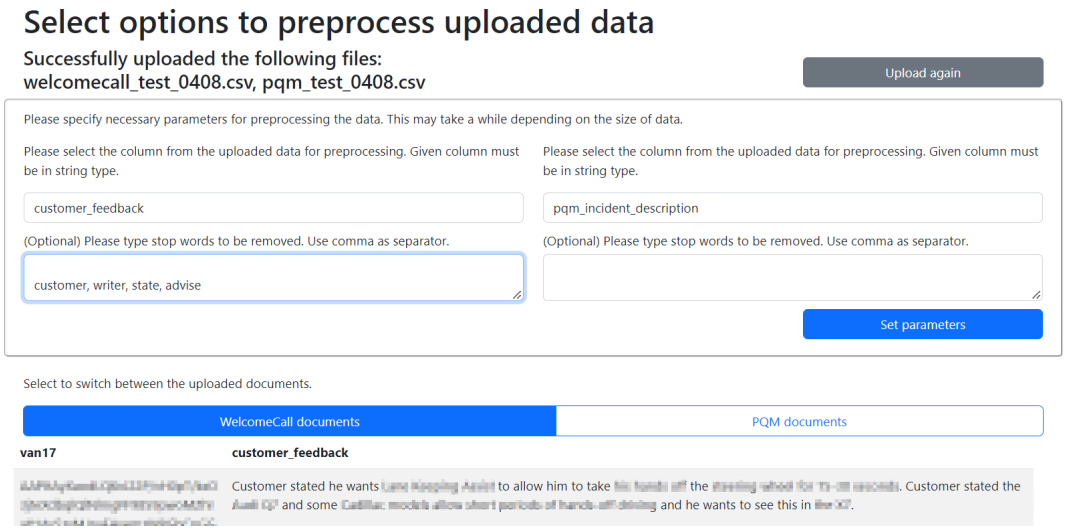


Figure 6.11.: Preprocessing page

Prediction Page

The output of the preprocessing step is used as input for this page. If the step is unsuccessful, it displays an error message as shown in Figure 6.12. If the preprocessing step is successful, a preview of cleaned documents that will be the input of the prediction end-point is displayed in a simple table view as shown in Figure 6.13. The table allows the user to double-check the documents before proceeding to the prediction step. By clicking on the 'Back to preprocessing' button user can revert to the preprocessing step, and by clicking on 'Predict documents', the user can generate topic predictions for the uploaded documents. This action redirects the user to the 'View Documents' tab, whether the prediction is successful or not. If the prediction is not successful, system documents are displayed.

As part of the prediction process, the document frequencies for each topic are updated in the log files after the prediction is completed. These log files are used as source to the tables displayed in 'Table View' (subsubsection 6.1.3) and 'Topic details' pages (subsection 6.1.4). To revert the prediction and reset the system to static documents, the user can click on the 'Clear Documents' button, which is only available if new data is uploaded.

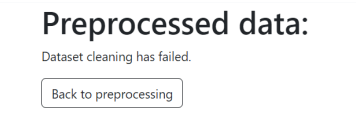


Figure 6.12.: Error message after unsuccessful preprocessing step

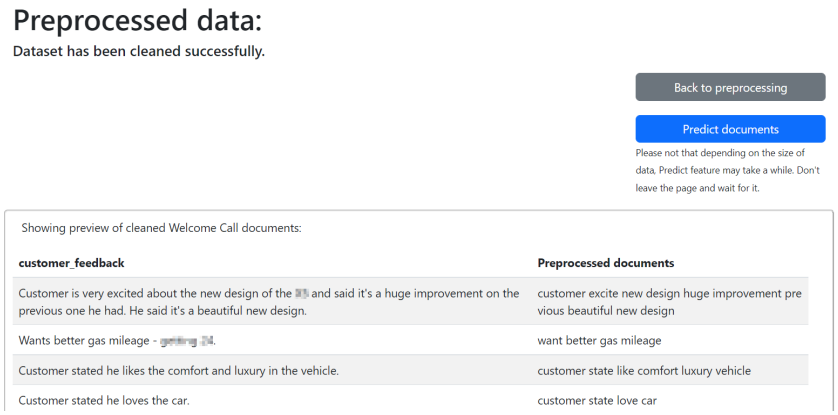


Figure 6.13.: Predict page

6.2.2. Features

Similarity Tables

Similarity logs that have been generated during the experiments (see subsubsection 5.2.1) are used to fill the table, with the addition of document frequency information. Document frequency is a built-in feature of BERTopic, which saves the information of document count per topic as part of the model during training. This frequency information has been concatenated to the similarity tables on the back-end and is updated as part of the topic prediction if the user has uploaded new documents to the system.

Heatmap

For heatmap implementation, we used Plotly [91]. Plotly [2] is an interactive, open-source, and browser-based graphing library that has implementations for both Python and JavaScript.

[2]https://github.com/plotly/plotly.py

Heatmap takes a 2-dimensional matrix as input; in this case, it is the distance matrix calculated through similarity measures as explained in section 5.2, that consists of the similarity scores between the topics of models. We stored the distance matrix as a static log file; however, if the file is not found, it is generated through Cosine Similarity calculation between topic embeddings of two models. The heatmap is generated inside the Python environment, converted to a JSON file, and then visualized with the JavaScript implementation of the Plotly library.

WordCloud

For the 'General Comparison' and 'Topic details' pages, Word Cloud images are generated. For each page, when a page is loading, it checks if the Word Cloud image file for the corresponding topic exists in the file system; if the file exists, it retrieves the file path and displays the image. If the file does not exist, it uses the WordCloud library to generate a new Word Cloud file for the given topic. WordCloud [3] is a Python library given a word-frequency pair that visualizes the frequency of the terms for text data which is the generated topic words in this case. Topic pair matches generated during similarity calculations are used to visualize the word cloud pairs side by side for the 'General Comparison' page.

Document Tables

These tables are displayed on the 'View Documents' and 'Topic details' pages. Depending on the state of the system, which is displayed on the page footer, four different data are used as the main data to be shown in these tables (1) static system documents, (2) uploaded documents, (3) preprocessed documents, (4) predicted documents.

Column Name	Description
Entry id	van 17 for Welcome Call documents or PQM problem number for PQM documents (text)
Topic id	Predicted topic id for the document entry. (numeric)
Topic words	Topic words for given topic id. (text)
Original Documents	Documents that are used as input for predictions. (numeric)

Table 6.2.: Document table column metadata

The column information is collected from the data files for (2) uploaded and (3) preprocessed documents. Table 6.2 explains the metadata of the columns for the table that is displayed with (1) static system documents and with (4) predicted documents.

[3] https://github.com/amueller/word_cloud

7. Evaluation and Discussion

7.1. User Survey

In order to evaluate the system we have built, we used a questionnaire. The system was tested by BMW employees who are competent in the area of production as well as knowledgeable on customer feedback dataset. The questionnaire consists of fourteen structured statements to obtain the participant's agreement level, scaled from 1 (Strongly Disagree) to 5 (Strongly agree), as well as two statements of unstructured text entries, which allows participants to input their own opinion. We separated the questions into four categories.

7.1.1. Model Comparison

This part of the survey evaluates the results of generated model comparisons and topic matching between the datasets. It has three statements on the Table View, Heatmap View, and Word Cloud View from the 'General Comparison' page of the web application.

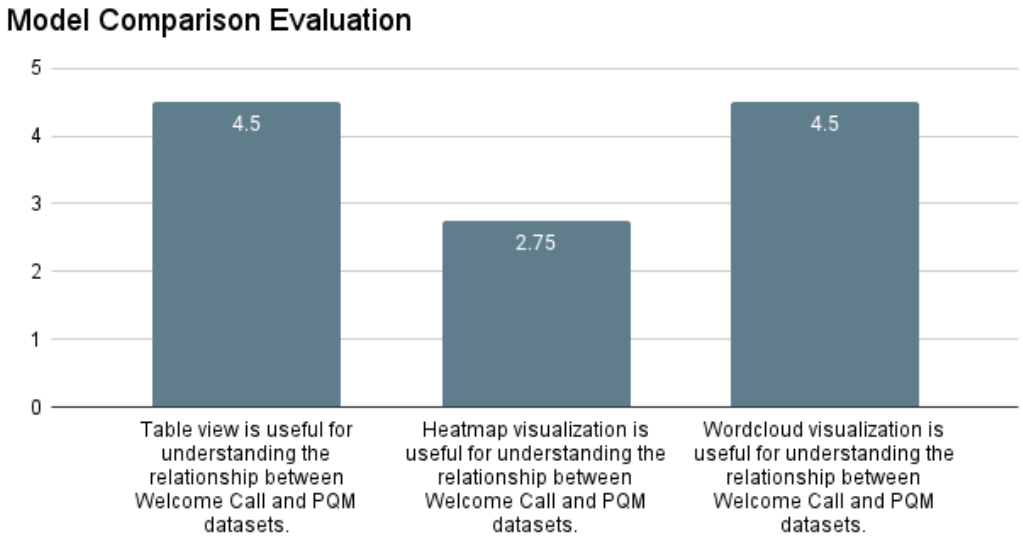


Figure 7.1.: A summary of feedback on Model Comparison Evaluation. Each value on the x-axis displays the statements that were asked of participants for rating. The y-axis shows an average of the gathered scores, on a 1-5 scale, with 1 meaning Strongly Disagree and 5 meaning Strongly Agree.

We chose to take the average of the votes to compare respective views on the page, shown in Figure 7.1. The participants found the Table View and the Word Cloud View equally helpful in understanding the relationship between datasets, whereas the Heatmap view had mixed reviews. Some users thought the heatmaps were very useful, whereas some found them confusing.

7.1.2. Result Quality

Six statements were asked of participants to rate the quality of the results generated through the topic modeling and topic matching pipelines. For each dataset, participants are asked to rate the quality of topic words generated by the topic models in two statements and the document-wise topic predictions in two statements. The other two statements are asked to determine the accuracy of similarity matching between models and whether the system helped understand the relationship between the issues occurring in datasets. The generated topic words of Welcome Call and PQM models made sense to most users, with some of them answering as neutral to this statement shown in Figure 7.2 and Figure 7.3. Document-wise topic predictions are generally perceived sensible for both Welcome Call and PQM models, shown in Figure 7.4 and Figure 7.5. Figure 7.6 shows that all participants found the similarity matching between models accurate; they also found the system useful for understanding the relationship between Welcome Call and PQM issues shown in Figure 7.7.

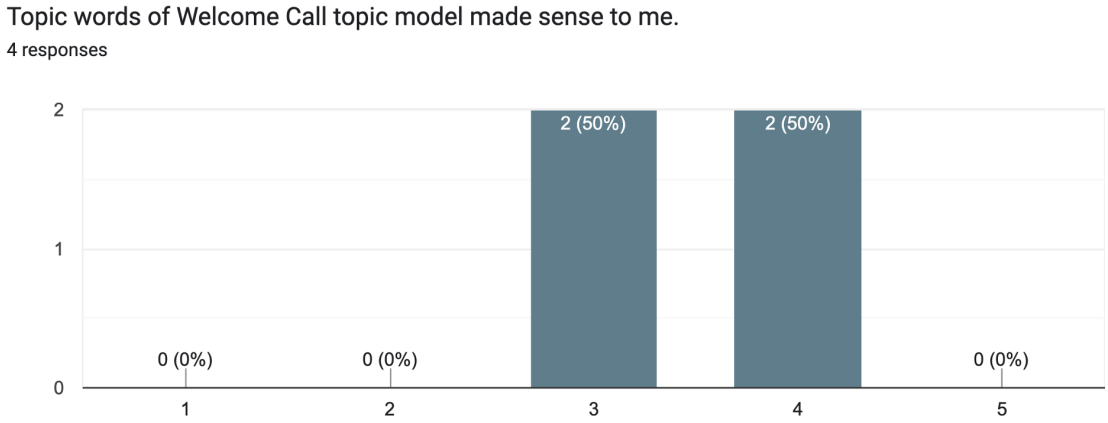


Figure 7.2.: A histogram of feedback from participants on the quality of topic words generated by the Welcome Call model. The x-axis shows the votes on a 1-5 scale, with 1 meaning Strongly Disagree, and 5 meaning Strongly Agree, and the y-axis shows the number of votes the statement received.

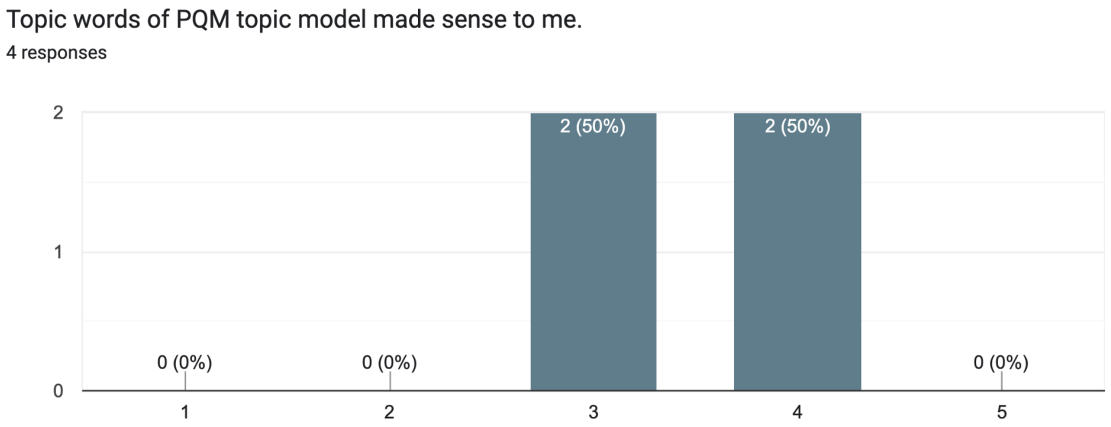


Figure 7.3.: A histogram of feedback from participants on the quality of topic words generated by the PQM model. The x-axis shows the votes on a 1-5 scale, with 1 meaning Strongly Disagree, and 5 meaning Strongly Agree, and the y-axis shows the number of votes the statement received.

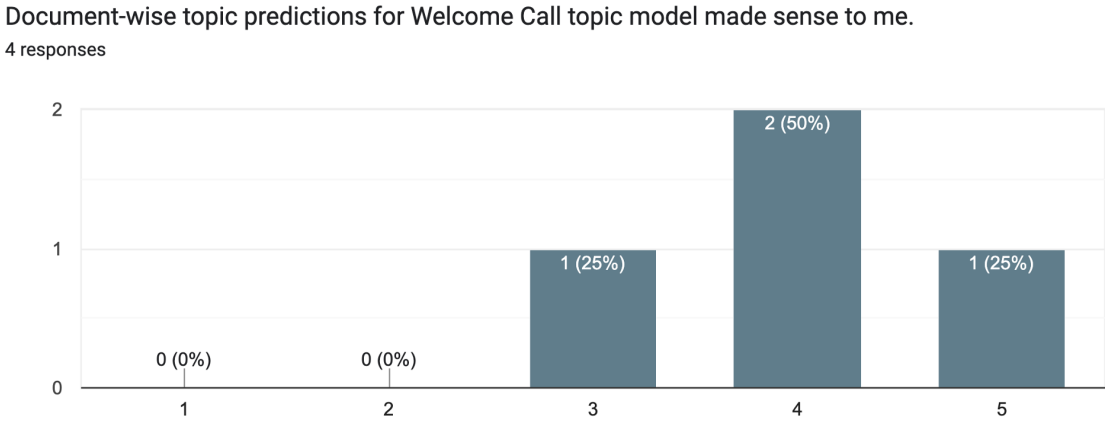


Figure 7.4.: A histogram of feedback from participants on document-wise prediction quality of Welcome Call model. The x-axis shows the votes on a 1-5 scale, with 1 meaning Strongly Disagree, and 5 meaning Strongly Agree, and the y-axis shows the number of votes the statement received.

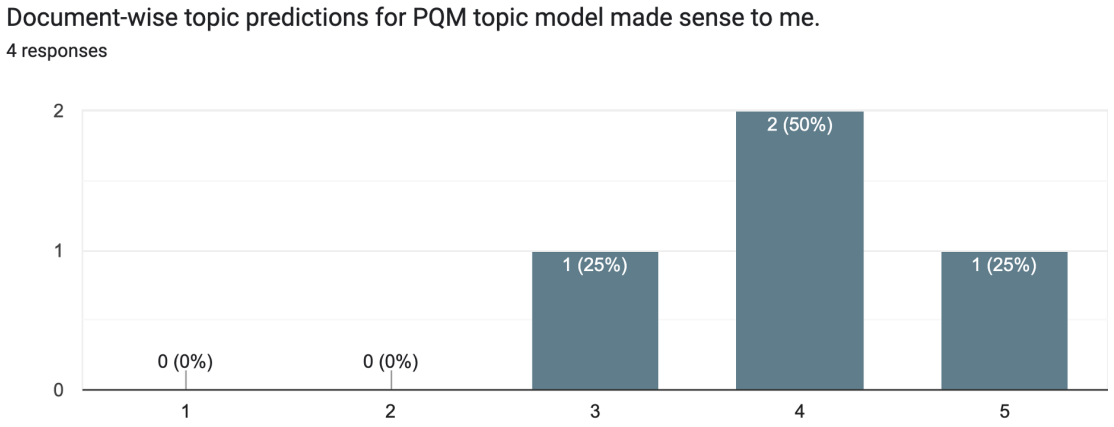


Figure 7.5.: A histogram of feedback from participants on document-wise prediction quality of PQM model. The x-axis shows the votes on a 1-5 scale, with 1 meaning Strongly Disagree, and 5 meaning Strongly Agree, and the y-axis shows the number of votes the statement received.

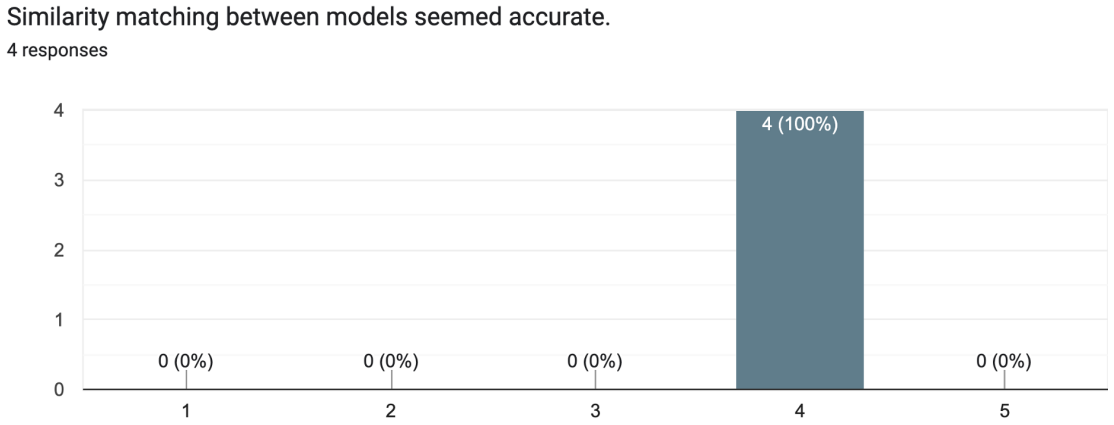


Figure 7.6.: A histogram of feedback from participants on the quality of the similarity matching. The x-axis shows the votes on a 1-5 scale, with 1 meaning Strongly Disagree, and 5 meaning Strongly Agree, and the y-axis shows the number of votes the statement received.

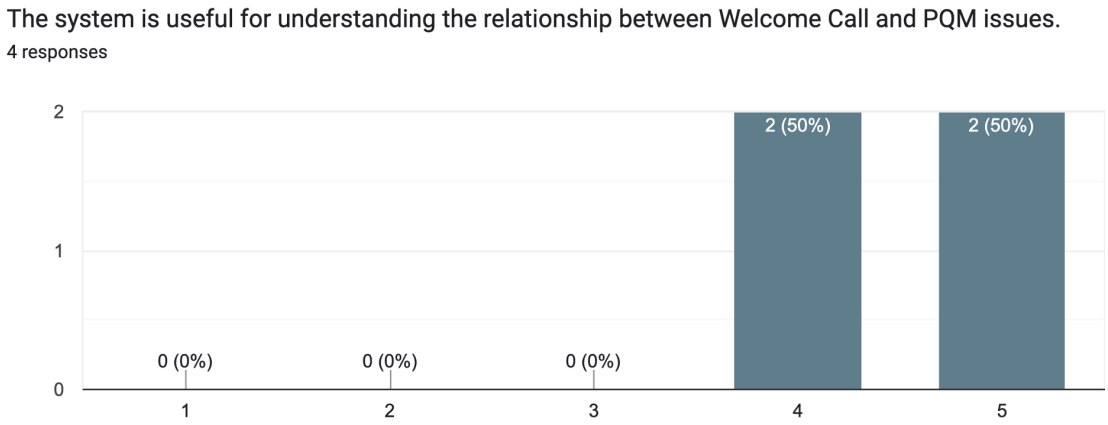


Figure 7.7.: A histogram of feedback from participants on the understandability of the relationship between the issues occurring in the datasets. The x-axis shows the votes on a 1-5 scale, with 1 meaning Strongly Disagree, and 5 meaning Strongly Agree, and the y-axis shows the number of votes the statement received.

7.1.3. User Interface Quality

User interface quality is assessed with three statements on the website's usability, appearance, and loading speed. As shown in Figure 7.8, the participants found the website easy to use and navigate. The system is observed to be displaying results quickly.

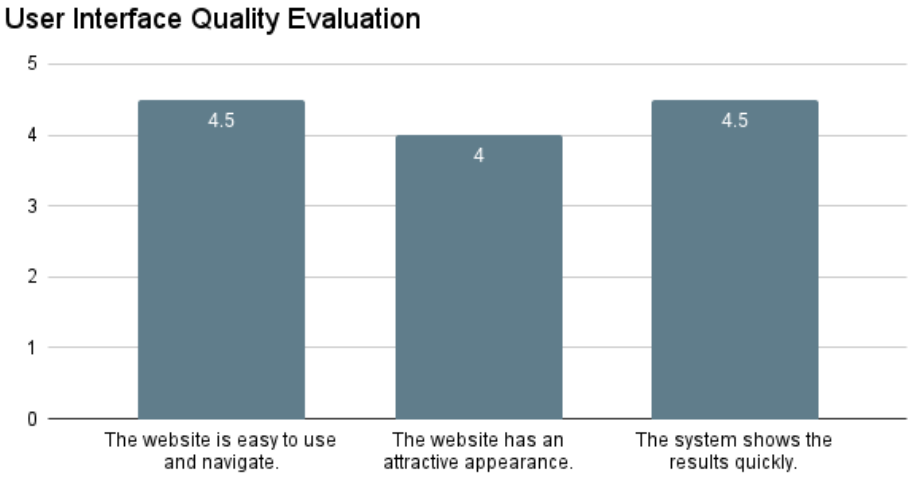


Figure 7.8.: A summary of feedback on User Interface Evaluation. Each value on the x-axis displays the statements that were asked of participants for rating. The y-axis shows an average of the gathered scores on a 1-5 scale, with 1 meaning Strongly Disagree and 5 meaning Strongly Agree.

7.1.4. Recommendations

The participants were asked whether they would use the framework and whether they would recommend the framework to other colleagues from a similar domain in two statements. They were also asked what they liked about the system and what they would want to see in the future as an improvement to the system. Figure 7.9 shows that all of the participants said they would use the framework themselves, and Figure 7.10 shows that except for one participant, all said they would recommend the system to their colleagues.

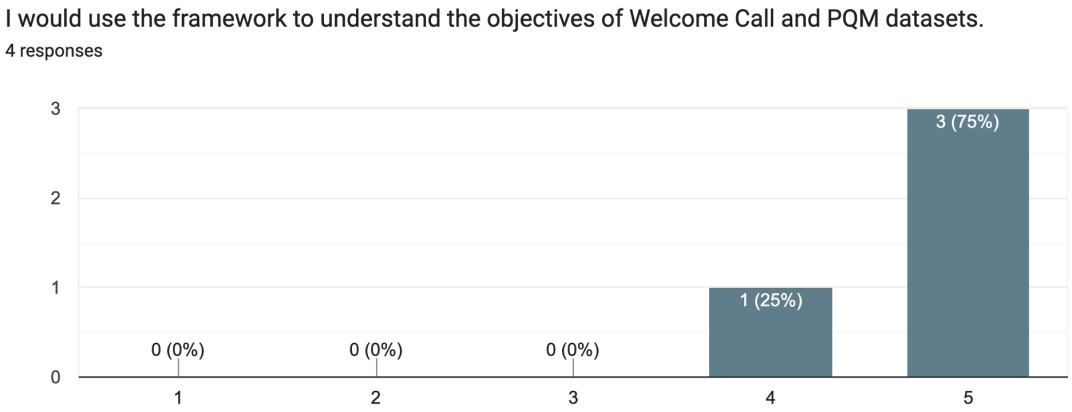


Figure 7.9.: A histogram of feedback from participants on the usage of the framework. The x-axis shows the votes on a 1-5 scale, with 1 meaning Strongly Disagree, and 5 meaning Strongly Agree, and the y-axis shows the number of votes the statement received.

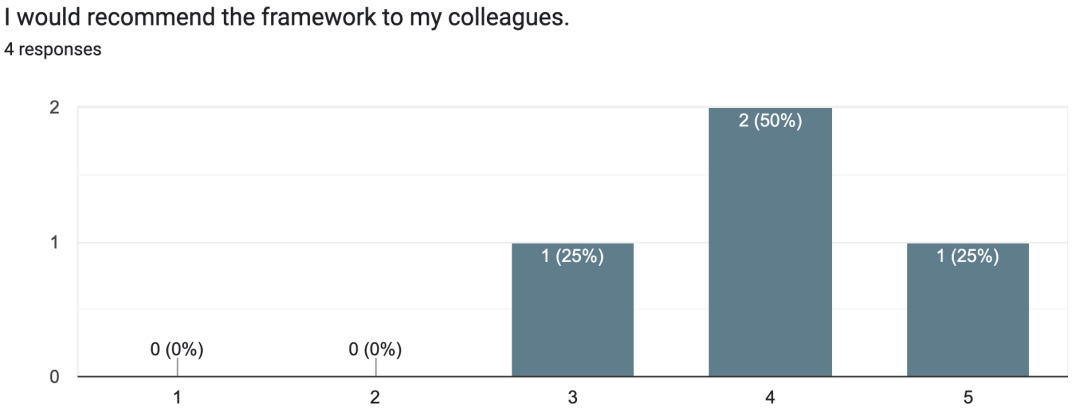


Figure 7.10.: A histogram of feedback from participants on the recommendation of the framework. The x-axis shows the votes on a 1-5 scale, with 1 meaning Strongly Disagree, and 5 meaning Strongly Agree, and the y-axis shows the number of votes the statement received.

7.2. Research Questions

In section 1.4, we have defined our research questions. Based on the results of our experiments explained in chapter 5 and the user study results, we will now give answers to our research questions.

Which state-of-the-art topic modeling approaches would provide better insight into BMW customer feedback datasets?

Embedding-based approach BERTopic provides better insights into BMW customer feedback datasets than traditional approach LDA. BERTopic generates more domain-specific topics, whereas LDA results in more general topic representations.

Which text similarity techniques give better matches between BMW customer feedback datasets?

Cosine Similarity with contextual embeddings (Sentence-BERT) provided better topic matches over Soft Cosine Similarity with pretrained word embeddings (GloVe).

How to support quality control departments with interactive topic visualizations of BMW customer feedback datasets?

Intertopic distance maps were found useful when selecting which topic model is better; however, when presented to domain experts, they were considered confusing. The user study showed that viewing the topic matches in a simple table format and word cloud representations are equally helpful. Heatmap visualization had a mixed reaction from the users; some users found it very useful, whereas some discussed that it was confusing.

8. Limitations and Future Work

In this study, we worked with topic modeling approaches with different hyperparameter settings. The results we gathered have shown to be helpful in analyzing and gaining insights from customer feedback datasets. We have shown that the topics generated by BERTopic are more meaningful than the ones generated by LDA. However, one advantage of LDA is that one document can contain multiple topics, which is not possible with BERTopic. It could have been beneficial to use embedding-based approaches and be able to retrieve multiple topics occurring in one document if there are any. However, our literature research showed that embedding-clustering-based topic modeling methods are not able to accomplish this. We believe that this issue could be successfully overcome with the proceedings of new research.

Due to the time limitation of this study, we did not get to explore other contextual embeddings for BERTopic implementation. BERTopic supports various embedding models to generate document embeddings, which could be explored to create a comparative study on the application of contextual embeddings on topic modeling.

Apart from BERTopic and LDA, we have also tried implementing the Top2Vec [5] method; however, we have encountered a problem with their implementation on UMAP, which was limiting us. Using more than 112K samples gives an error, which was not suitable for our case since both of our datasets were double that size [1]. If this problem gets resolved, a comparison between embedding-based topic models could be accomplished. Another topic modeling library, TopClus [53], was published after we started our work on this study, which has reported better results than BERTopic; it would be interesting to see how this model works on datasets within the production domain.

We wanted to explore other visualization methods such as word networks and word bubbles; however, it was decided that word networks were a complex visualization that is not as easy to understand as word clouds for users unfamiliar with the NLP domain. Word bubbles are a more promising visualization method; however, we were unable to find an implementation in Python. We have found an implementation with D3 [2], which uses JavaScript, that could be used to improve the web application we implemented.

Another limitation we encountered was the implementation of the web application. BMW Group already has well-established BI tools to store and analyze the datasets. However, we were not able to integrate our models inside the BI tool since it has limited support for machine learning models. The feedback we gathered from our user study showed that integrating the topic models into the internal BI tool would have been helpful to automate the upload of the data. We believe that our tool can be integrated into the internal BMW BI tools in the future.

[1] https://github.com/ddangelov/Top2Vec/issues/215
[2] https://d3js.org/

9. Conclusion

This study has aimed to discover new production issues by exploring the BMW customer feedback dataset (Welcome Call) and comparing the results with the BMW vehicle production incidents dataset (PQM). Topic modeling and text matching techniques have been the research focus of this study, and a Flask-based[11] web application is designed as the final product of this study.

For topic modeling experiments, LDA [2] as the traditional topic modeling approach, and BERTopic [10] as the embedding-based approach were used, testing different topic cluster numbers and topic word numbers to find an appropriate representation for the provided datasets. To inspect the experiments, intertopic distance maps, retrieved topic words, and topic coherence metrics (*Umass* [81]) were used. Intertopic distance maps are generated with pyLDAvis [90] for LDA models and built-in implementation is used for BERTopic models. It was observed through intertopic distance maps that between 75-100 topic clusters provide a good representation for both of the datasets and both topic modeling approaches. However, it was difficult to say which setup was better. As for the number of topic words parameter, it was observed that even though the models with top-20-words contain more noise, they give more information on the topic words than the top-10-words models, which have higher topic coherence scores. Topic word retrieval showed that LDA models provided semantically more general topic representations, whereas, with BERTopic models, we were able to identify the production-relevant issues better. Word Clouds were generated from the topic words to visualize the outputs of topic models in a more user-friendly way.

For text-matching experiments, Cosine Similarity was applied to topic embeddings generated through Sentence-BERT, whereas for Soft Cosine Similarity, fixed word embeddings of GloVe[17] were used. BERTopic models with 80, 85, 90, and 95 topic clusters, with top-10 topic words and top-20 topic words, were used as input for the similarity calculations. Similarities for the top-10-words models and the top-20-words models were calculated separately for a fair comparison. For each model pair, topic-based similarity scores were calculated and stored in a table to determine the matching topics between datasets and to identify the new topics occurring in the customer feedback dataset. The resulting topic similarities were used to generate a heatmap and were classified into two labels, 'Match' and 'Non-Match', by applying a threshold to find the matches. To evaluate the match quality, topic matches were inspected and labeled manually. Precision, Recall, and F1 Score were reported as results. Among Welcome Call models the 85 topic clusters model, and among PQM models the 95 topic clusters model were selected to be used in the web application. For the chosen model pair, 47 new issues were discovered among 85 Welcome Call topics.

A user study with participants from the BMW Group was conducted to evaluate the designed web application. The overall feedback on the tool was positive; the participants

stated they would use the system and also recommend it to their colleagues. The user interface was reported as attractive and easy to use, plus the back-end system was considered quick. The participants stated that they found the similarity tables and word cloud representations the most useful, whereas heatmap visualization received mixed reviews. The similarity matching of topics and the topic words generated by the models were found accurate by the participants, and the prediction capabilities of the topic models received a good impression.

A. Tables

Number of topic clusters	Keyword	Topic words
80	brake	brake, leak, line, inspection, find, pad, axle, pedal, fill, vehicle, visual, test, scan, vacuum, sign, area, check, aux, evident, rig
	seat	seat, rear, leather, backrest, windscreen, cushion, row, crease, squab, specify, forward, blind, position, diameter, fasten, area, hume, wrinkle, middle, lhf
	mirror	water, mirror, ingres, seal, flange, cause, test, area, boot, exterior, spot, space, find, corrosion, sealer, fzd, vehicle, drip, dust, biw
85	brake	brake, pad, pedal, fluid, remain, visual, meldung, vacuum, previously, caliper, disc, whine, cabin, line, hydraulic, depend, alex, vehicle, flock, heckklappe
	seat	seat, rear, belt, soft, row, manual, play, buckle, fog, rotate, audi, protective, en, app, function, kann, usche, attachment, matrix
	mirror	mirror, cad, actual, movement, tolerance, direction, martin, cobblestone, exterior, robert, fzd, vfc, branch, data, axis, higher, jack, frt, overlap, nominal
90	brake	brake, pad, pedal, relay, vacuum, stall, hinten, ambient, caliper, dvd, rechts, fluid, ca, vehicle, park, ef, arbeiten, durchgef, shifter, vorne
	seat	seat, rear, leather, middle, backrest, cover, row, crease, squab, area, tilt, driver, song, kidney, inboard, outboard, sufficient, issue, leg, attachment
	mirror	mirror, heat, flap, filler, exterior, heater, hall, fuel, thomas, noisy, mi, stage, external, mix, subject, schmidt, permanently, notify, attachements, vision
95	brake	brake, pedal, usa, play, manual, feel, die, nr, vacuum, performance, free, intermittent, ram, device, ri, ag, booster, system, werden, adjustable
	seat	seat, rear, backrest, airbag, instrument, cushion, cluster, long, lhr, leave, bolster, extend, position, area, leather, forward, visible, right, lower, issue
	mirror	mirror, stick, gear, flap, shift, qzs, operate, jam, lever, underbody, functional, collision, sport, cycle, con, september, brown, kidney, vibrate, selector

Table A.1.: Per each number of topic clusters, topic word examples generated by LDA model, for PQM data, range 80-95

Number of topic clusters	Keyword	Topic words
50	brake	vehicle, service, time, dealership, happen, bmw, notice, car, follow, brake, drive, week, issue, come, mile, day, work, center, bring, appointment
	seat	seat, like, passenger, belt, position, want, adjust, driver, lower, rear, adjustment, comfortable, row, leather, sit, able, support, stick, easily, document
	mirror	mirror, windshield, fold, wiper, wash, rain, prompt, exterior, specific, automatically, passenger, mail, want, position, emergency, like, lighter, automatic, car, sensor
75	brake	brake, windshield, wiper, weather, build, fully, rain, vehicle, moment, apply, floor, read, speedometer, gauge, mat, fast, fog, like, snow, rubber
	seat	seat, like, passenger, belt, want, heat, sun, area, higher, leather, sit, easily, material, able, inch, visor, adjustment, easy, acceleration, smoother
	mirror	mirror, automatically, inside, fold, view, stick, switch, like, vehicle, passenger, want, curb, soft, exterior, position, manually, bar, lift, able, countryman
100	brake	brake, vehicle, higher, highway, pick, mph, drive, pad, speed, felt, apply, squeak, mile, manager, stop, normal, visibility, dealership, slow, experience
	seat	seat, passenger, heat, like, belt, rear, deactivate, want, row, leather, driver, curb, able, leg, buckle, position, massage, second, extender, warmer
	mirror	screen, mirror, idrive, rattle, inside, easily, like, maybe, pa, write, want, able, mid, newest, vehicle, use, currently, denver, downtown, pronounce
125	brake	brake, point, temperature, delivery, vehicle, ventilate, release, couple, complete, scratch, seatbelt, notice, catch, rust, car, ask, know, bmw, depart, look
	seat	seat, like, choose, want, sensitive, able, child, universal, massage, jump, transmitter, way, aspect, car, task, try, multi, desire, master, contour
	mirror	mirror, fold, exterior, adjustment, curb, passenger, like, want, enjoy, gps, portion, effect, column, product, vehicle, distract, chrome, reflect, flash, solve
150	brake	brake, break, vehicle, mile, noise, drive, notice, symptom, dealership, replicate, service, rougher, incline, ask, little, bmw, use, time, check, westchester
	seat	seat, passenger, like, adjustment, leather, split, sit, maximum, rub, option, forward, vehicle, seam, notice, ask, use, try, time, know, right
	mirror	mirror, heat, position, utilize, tilt, curb, passenger, resource, slide, like, automatic, briefly, ask, backrest, distribution, vehicle, able, september, use, look
250	brake	brake, vehicle, squeak, noise, louder, apply, decide, impact, pad, mitigation, unpleasant, tape, slower, dealership, axle, squeal, come, time, need, caliper
	seat	seat, additional, adjustment, row, encourage, need, notify, able, vehicle, dealership, service, ownership, help, plan, begin, bring, day, away, stick, appreciate, include, movement, tell, similar, preventitive, prevelant
	mirror	mirror, fold, exterior, curb, like, vehicle, external, ontario, unfold, rental, sure, time, car, driver, set, button, help, activate, bmw, door

Table A.2.: Per each number of topic clusters, topic word examples generated by LDA model, for Welcome Call data, range 50-250

Number of topic clusters	Keyword	Topic words
80	brake	brake, break, vehicle, mile, case, pad, exhaust, performance, period, stop, apply, car, squeak, drive, gallon, normal, noise, slow, opinion, inform
	seat	seat, rear, passenger, position, like, belt, item, want, able, sense, angle, meet, leg, enjoy, way, tighten, find, place, heat, lexus
	mirror	mirror, right, hand, fold, leave, exterior, difficulty, quickly, easily, passenger, curb, proper, corner, switch, like, want, automatic, reverse, catch, know
85	brake	brake, noise, vehicle, drive, stop, apply, squeak, mile, hear, sound, come, dealership, tab, service, concern, notice, grey, bmw, actual, time
	seat	seat, like, passenger, cruise, position, belt, limit, adjustment, comfortable, want, adaptive, sit, vehicle, driver, able, forward, find, silver, lumbar, height
	mirror	mirror, hard, fold, miss, exterior, tilt, welcome, like, passenger, smooth, underneath, finally, vehicle, previous, external, right, generation, view, look, near
90	brake	brake, high, vehicle, stop, engage, apply, squeak, tone, audible, beam, speed, mile, accelerator, portion, normal, drivetrain, noise, pad, worse, resistance
	seat	seat, automatic, like, limit, rear, want, package, adjustment, faster, confuse, quickly, able, chance, kind, corner, come, way, crash, rub, executive
	mirror	mirror, passenger, fold, driver, exterior, water, curb, split, monitor, like, want, position, friend, actual, able, rock, odd, company, clarify, vehicle
95	brake	brake, noise, sound, hear, vehicle, come, drive, quality, build, like, speed, bump, apply, squeak, service, notice, road, dealership, speaker, bmw
	seat	seat, adjust, like, belt, lower, want, adjustment, comfortable, row, height, passenger, higher, driver, able, heat, rear, uncomfortable, way, raise, buckle
	mirror	mirror, fold, reverse, exterior, satellite, tilt, curb, passenger, monitor, morning, want, like, maintenance, winter, automatic, randomly, typically, vehicle, driver, metal

Table A.3.: Per each number of topic clusters, topic word examples generated by LDA model, for Welcome Call data, range 80-95

Number of topic clusters	Keyword	Topic ID	Topic words
80	brake	11	brake, noise, squeak, dust, break, squeal, period, mile, grind, stop, rotor, pad, normal, noisy, hear, sound, dealership, notice, feel, loud
		65	collision, warn, brake, frontal, mitigation, red, pedestrian, alert, system, safety, forward, apply, imminent, feature, approach, avoidance, speed, intervention, accident, stop
	seat	38	seat, row, massage, room, rear, adjustment, leg, passenger, adjust, space, recline, message, second, lock, interior, position, forward, chair, like, child
		49	ventilate, heat, seat, climate, sync, control, cool, contact, temperature, air, hot, ventilation, button, rule, synchronize, row, passenger, like, want, setting
		62	leather, seat, support, lumbar, thigh, extender, leg, stitch, passenger, picture, wrinkle, piece, upholstery, plastic, driver, loose, material, dealership, extend, send
	mirror	19	fold, mirror, curb, tilt, passenger, monitor, reverse, automatically, exterior, lock, automatic, switch, position, seat, driver, view, door, feature, unfold, deactivate
85	brake	4	brake, noise, squeak, collision, dust, break, stop, squeal, period, mile, warn, normal, pad, rotor, noisy, frontal, hear, sound, apply, speed
		74	launch, brake, pedal, traction, auto, control, hold, accelerator, dsc, park, stability, dynamic, flag, foot, press, button, engage, snow, release, ask
	seat	43	seat, row, massage, room, rear, leg, adjustment, recline, passenger, second, space, message, lock, interior, adjust, position, chair, forward, fold, child
		46	heat, ventilate, climate, seat, sync, control, temperature, cool, contact, air, button, rule, synchronize, ventilation, set, hot, passenger, setting, row, want
		48	leather, seat, interior, upholstery, jean, material, stitch, picture, stain, look, passenger, wrinkle, cognac, clean, plastic, quality, color, notice, vernasca, dealership
		62	seat, memory, save, position, profile, driver, set, mirror, fob, setting, key, lock, number, adjust, button, exit, wife, automatically, power, use
		66	seat, support, lumbar, thigh, comfortable, uncomfortable, extender, leg, bolster, cushion, like, wider, adjustable, sport, lever, little, adjust, feel, softer, extend
		77	belt, seat, buckle, seatbelt, tighten, passenger, neck, uncomfortable, height, fasten, latch, adjust, adjustable, warn, far, difficult, easier, hard, tight, like
	mirror	14	fold, mirror, curb, tilt, passenger, monitor, reverse, automatically, exterior, lock, automatic, switch, position, driver, seat, view, feature, door, deactivate, unfold
		56	android, mirror, screen, auto, entertainment, apple, available, iphone, video, carplay, compatible, want, watch, like, rear, phone, use, software, iphones, device
90	brake	11	brake, noise, squeak, dust, break, squeal, period, mile, stop, pad, rotor, normal, noisy, hear, sound, notice, dealership, feel, loud, service
		39	brake, collision, launch, warn, pedal, frontal, auto, hold, control, mitigation, accelerator, park, red, engage, foot, flag, pedestrian, feature, stop, press
	seat	29	heat, ventilate, climate, seat, temperature, sync, control, cool, air, contact, button, rule, hot, ventilation, synchronize, set, setting, passenger, like, want
		60	leather, seat, support, thigh, lumbar, extender, leg, stitch, passenger, picture, wrinkle, piece, upholstery, plastic, driver, material, loose, notice, look, send
		66	belt, seat, buckle, seatbelt, tighten, passenger, neck, uncomfortable, fasten, height, adjust, latch, hard, adjustable, warn, far, difficult, easier, retract, pillar
		78	seat, row, child, rear, adjustment, lock, adjust, recline, second, message, passenger, position, chair, captain, forward, fold, door, safety, backrest, open
	mirror	21	mirror, curb, android, tilt, screen, monitor, passenger, reverse, auto, entertainment, switch, automatic, available, position, feature, deactivate, iphone, downward, want, know
		49	fold, mirror, automatically, lock, exterior, seat, unfold, rear, row, door, cargo, captain, want, button, access, chair, like, setting, manually, set
95	brake	13	brake, noise, squeak, dust, break, squeal, period, grind, mile, stop, pad, normal, rotor, noisy, hear, sound, dealership, notice, loud, feel
		61	collision, warn, brake, frontal, mitigation, red, pedestrian, alert, system, safety, forward, approach, imminent, apply, avoidance, speed, feature, accident, intervention, early
	seat	20	seat, ventilate, massage, heat, cool, air, condition, standard, like, wish, rear, want, option, hot, package, stronger, feel, ventilation, warmer, feature
		50	seat, row, room, rear, adjustment, leg, recline, space, second, adjust, message, lock, interior, passenger, position, chair, forward, fold, child, captain
		59	leather, seat, support, thigh, lumbar, extender, leg, stitch, picture, passenger, wrinkle, upholstery, plastic, piece, material, driver, loose, send, extend, notice
		72	belt, seat, buckle, seatbelt, tighten, neck, passenger, uncomfortable, height, adjust, adjustable, fasten, far, latch, difficult, hard, easier, warn, like, sit
	mirror	11	mirror, fold, curb, tilt, passenger, monitor, screen, automatically, reverse, exterior, lock, automatic, switch, entertainment, rear, position, seat, feature, want, view

Table A.4.: Per each number of topic clusters, topic word examples generated by BERTopic model, for Welcome Call data, range 80-95

Number of topic clusters	Keyword	Topic ID	Topic words
50	brake	19	brake, noise, squeak, collision, stop, break, dust, mile, squeal, period, warn, normal, pad, rotor, vehicle, hear, sound, noisy, apply, frontal
	seat	24	heat, seat, ventilate, wheel, steer, massage, cool, like, standard, want, climate, hot, rear, button, control, air, warm, wish, temperature, option
	mirror	14	mirror, fold, android, curb, tilt, passenger, screen, auto, monitor, reverse, automatically, exterior, lock, automatic, want, switch, seat, rear, like, feature
75	brake	45	brake, noise, squeak, dust, squeal, break, mile, period, pad, rotor, normal, noisy, hear, sound, stop, loud, dealership, notice, service, grind
	seat	28	leather, interior, seat, color, upholstery, material, look, stitch, beige, black, jean, like, picture, stain, plastic, white, passenger, cognac, red, clean
	mirror	20	fold, mirror, curb, tilt, passenger, monitor, reverse, automatically, exterior, lock, automatic, switch, position, seat, driver, view, feature, door, deactivate, button
100	brake	27	brake, noise, squeak, dust, squeal, break, rotor, mile, period, pad, normal, noisy, hear, sound, loud, stop, notice, dealership, grind, service
	seat	41	belt, seat, buckle, seatbelt, tighten, neck, passenger, uncomfortable, height, adjust, fasten, adjustable, latch, hard, retract, far, easier, difficult, unbuckle, like
		58	leather, seat, support, thigh, lumbar, extender, leg, stitch, picture, passenger, wrinkle, piece, upholstery, plastic, material, loose, extend, driver, send, look
	mirror	16	fold, mirror, screen, automatically, lock, exterior, entertainment, rear, seat, android, want, unfold, iphone, video, like, door, row, able, watch, cargo
125	brake	14	brake, noise, squeak, dust, squeal, grind, rotor, break, pad, period, noisy, normal, mile, hear, sound, loud, stop, dealership, notice, performance
	seat	44	heat, massage, seat, climate, rear, temperature, control, rule, hot, standard, cool, row, armrest, warm, ventilation, passenger, want, like, wish, stronger
		46	leather, seat, support, thigh, lumbar, extender, leg, stitch, picture, passenger, wrinkle, piece, upholstery, plastic, material, loose, extend, driver, send, uncomfortable
	mirror	13	fold, mirror, screen, automatically, lock, exterior, android, seat, unfold, rear, iphone, video, want, row, watch, cargo, captain, entertainment, door, like
150	brake	17	brake, noise, squeak, dust, squeal, rotor, break, pad, period, noisy, normal, mile, hear, sound, loud, stop, grind, notice, dealership, performance
		96	brake, feel, sensitive, stop, regenerative, pedal, smooth, break, period, jerk, apply, power, little, previous, shudder, felt, touchy, harder, slow, like
	seat	50	seat, grind, comfortable, higher, uncomfortable, bolster, logo, height, cushion, sit, projector, raise, lower, little, low, like, adjustment, softer, projection, adjust
		52	belt, buckle, seat, seatbelt, tighten, neck, passenger, uncomfortable, height, fasten, adjustable, adjust, far, latch, tight, difficult, hard, easier, pillar, chime
	mirror	27	fold, mirror, automatically, lock, exterior, seat, unfold, row, rear, captain, cargo, chair, door, access, button, automatic, manually, want, flat, power
		97	mirror, screen, entertainment, android, video, iphone, watch, apple, rear, iphones, compatible, stream, hdmi, miracast, device, use, youtube, carplay, phone, able
250	brake	14	brake, squeak, noise, dust, squeal, rotor, break, noisy, pad, period, normal, mile, hear, loud, sound, grind, stop, performance, notice, axle
		56	brake, sensitive, feel, regenerative, stop, pedal, smooth, break, jerk, period, apply, shudder, park, little, power, touchy, regeneration, felt, previous, harder
	seat	33	comfortable, seat, uncomfortable, bolster, cushion, wider, softer, bucket, narrow, sport, feel, hard, little, pain, firm, leg, thigh, sit, adjustable, stiff
		64	memory, seat, save, position, profile, fob, driver, set, number, setting, key, exit, lock, wife, adjust, automatically, prefer, easier, button, husband
		73	leather, seat, stitch, wrinkle, picture, upholstery, thigh, loose, passenger, piece, undo, plastic, material, send, notice, support, delivery, merino, look, tear
	mirror	30	mirror, curb, tilt, reverse, monitor, passenger, position, switch, automatic, exterior, deactivate, downward, view, adjust, slide, driver, fold, right, disable, adjustment
		31	fold, mirror, automatically, exterior, lock, unfold, door, automatic, manually, access, power, view, setting, button, set, standard, want, fob, driver, close
		71	mirror, screen, entertainment, video, android, watch, iphone, iphones, hdmi, stream, compatible, rear, miracast, apple, chromecast, netflix, movie, youtube, device, support
		223	mirror, position, save, memory, profile, driver, setting, adjust, set, exterior, seat, fob, key, view, lock, tilt, adjustment, store, button, preference

Table A.5.: Per each number of topic clusters, topic word examples generated by BERTopic model, for Welcome Call data, range 50-250

Number of topic clusters	Keyword	Topic ID	Topic words
50	brake	11	brake, pedal, pad, clutch, pipe, caliper, fluid, dsc, disc, noise, park, vacuum, wear, line, vehicle, master, booster, customer, sensor, replace
	seat	0	seat, leather, backrest, rear, cushion, wavy, headrest, armrest, cover, crease, bolster, rest, audit, leave, issue, gap, right, lh, rh, visible
		9	belt, seat, seatbelt, buckle, retract, retractor, rear, rattle, safety, row, pillar, twist, receiver, passenger, right, find, trap, issue, noise, pull
		18	seat, noise, backrest, creak, armrest, squeak, adjust, headrest, static, click, rear, rest, hear, driver, adjustment, lean, forward, cushion, come, arm
	mirror	20	mirror, exterior, view, door, fold, switch, cap, leave, glass, right, outside, base, triangle, function, rear, noise, rearview, driver, cover, issue
75	brake	12	brake, pedal, pad, caliper, fluid, clutch, pipe, dsc, disc, wear, booster, master, vacuum, line, park, noise, accelerator, vehicle, sensor, servo
	seat	0	seat, leather, backrest, cushion, rear, headrest, wavy, armrest, crease, cover, bolster, rest, audit, leave, wrinkle, gap, issue, lh, right, visible
		9	belt, seat, seatbelt, buckle, retract, retractor, rattle, rear, safety, row, twist, pillar, receiver, passenger, trap, pull, right, issue, leave, find
		16	seat, noise, creak, backrest, squeak, armrest, adjust, headrest, hear, rest, click, driver, rear, arm, adjustment, lean, forward, cushion, passenger, center
	mirror	14	mirror, exterior, view, fold, door, glass, outside, cap, switch, leave, triangle, right, base, rearview, function, driver, noise, rear, cover, interior
100	brake	54	brake, pad, pedal, disc, caliper, park, noise, dsc, emf, squeal, wear, squeak, rotor, sensor, booster, speed, drive, vehicle, fluid, customer
		56	brake, pedal, clutch, fluid, accelerator, master, cylinder, vacuum, leak, booster, caliper, pad, bleed, wear, slave, servo, pipe, replace, test, find
		74	pipe, brake, pip, clip, air, line, touch, intake, filter, cleaner, fit, box, modulator, dsc, vacuum, servo, snorkel, clamp, duct, secure
	seat	0	seat, backrest, leather, cushion, headrest, rear, armrest, crease, bolster, wavy, rest, cover, audit, wrinkle, leave, row, visible, gap, issue, stitch
		6	belt, seat, seatbelt, buckle, retract, retractor, rattle, rear, safety, row, receiver, twist, passenger, pillar, trap, pull, reel, right, web, fit
		12	seat, noise, backrest, creak, squeak, armrest, headrest, adjust, rest, hear, click, driver, adjustment, lean, cushion, rear, forward, arm, passenger, weight
	mirror	9	mirror, exterior, view, fold, door, cap, triangle, switch, outside, glass, rearview, base, leave, right, noise, driver, function, cover, adjust, interior
		89	visor, sun, sunvisor, vanity, light, mirror, grab, vdc, fit, clip, driver, headliner, handle, hook, loose, inoperative, issue, wrong, hinge, pivot
125	brake	61	brake, pad, disc, noise, pedal, squeal, squeak, wear, caliper, speed, rotor, sensor, hear, light, apply, drive, stop, rear, disk, reverse
		62	brake, caliper, dsc, fluid, park, pipe, pad, emf, pip, line, pedal, disc, servo, master, actuator, booster, repair, unit, vacuum, clip
		75	pedal, brake, clutch, accelerator, fluid, pad, vacuum, cylinder, master, booster, slave, wear, bleed, cbs, depress, replace, leak, customer, sensor, servo
	seat	0	seat, leather, backrest, cushion, headrest, armrest, rear, crease, bolster, rest, wavy, cover, wrinkle, squab, audit, row, stitch, leave, foam, visible
		9	belt, seat, seatbelt, buckle, retract, retractor, safety, rear, row, twist, receiver, trap, pillar, pull, passenger, web, upper, fit, middle, anchor
		10	seat, noise, backrest, creak, armrest, squeak, headrest, adjust, static, click, rest, lean, hear, driver, adjustment, forward, mingxiang, cushion, passenger, rear
		36	heat, hd, seat, test, cycle, temperature, disassembly, disturbance, ptce, run, fslesen, damage, htoe, rain, lin, heater, rotor, button, detect, switch
		120	fan, electric, relay, cool, seat, overheat, fter, electrical, self, engine, elektrol, ventilation, cushion, run, breakdown, test, check, noise, diagnosis, dme
	mirror	12	mirror, exterior, view, fold, door, cap, outside, switch, rearview, triangle, glass, base, leave, right, function, driver, adjust, interior, noise, cover
		81	visor, sun, sunvisor, vanity, mirror, light, grab, vdc, hook, clip, driver, fit, headliner, inoperative, cover, function, makeup, handle, pivot, fix
150	brake	19	brake, pedal, fluid, clutch, caliper, pad, dsc, master, booster, vacuum, accelerator, cylinder, wear, servo, leak, bleed, disc, customer, pipe, replace
		32	spoiler, roof, tailgate, rear, gap, upper, brake, lower, lh, rh, profile, aeroblade, flush, active, condition, lid, paint, spec, light, allocation
		53	brake, pad, park, disc, pedal, noise, squeal, wear, squeak, caliper, emf, rotor, speed, sensor, hear, apply, release, drive, stop, switch
		135	brake, pipe, pip, clip, line, touch, servo, modulator, dsc, grommet, rhd, underbody, foul, fit, bulkhead, clash, firewall, brakelines, rout, hose
	seat	0	seat, backrest, leather, cushion, headrest, armrest, crease, rear, bolster, wavy, rest, cover, wrinkle, audit, squab, row, stitch, visible, leave, support
		10	seat, noise, backrest, creak, squeak, armrest, headrest, adjust, static, click, hear, rest, adjustment, mingxiang, driver, lean, cushion, forward, sunny, passenger
		11	belt, seat, seatbelt, buckle, retract, retractor, safety, twist, receiver, row, rear, trap, pillar, pull, web, middle, anchor, fit, loop, right
		25	heat, hd, seat, shield, cycle, test, temperature, disturbance, disassembly, ptce, run, heatshield, rotor, htoe, damage, fslesen, heater, lin, machine, mech
		102	rattle, belt, seat, retractor, noise, seatbelt, headrest, cobble, cobblestone, stone, buckle, surface, passenger, drive, reel, road, rear, hear, rest, faurecia
	mirror	12	mirror, exterior, view, fold, cap, door, triangle, switch, outside, glass, base, rearview, leave, right, function, driver, whistle, adjust, adjustment, interior
		71	visor, sun, sunvisor, vanity, mirror, light, grab, hook, vdc, driver, fit, clip, headliner, handle, inoperative, force, hinge, sunvisors, pivot, loose
		133	mirror, exterior, noise, fold, view, vibrate, vibration, squeak, door, rattle, rearview, close, hear, object, outside, creak, surface, road, interior, glass
250	brake	34	brake, pedal, fluid, accelerator, caliper, pad, master, booster, vacuum, wear, servo, dsc, leak, bleed, cbs, union, fill, cylinder, sensor, ab
		53	brake, pad, disc, squeal, pedal, noise, squeak, wear, caliper, rotor, speed, hear, apply, sensor, disk, ef, noisy, axle, reverse, slow
		104	brake, pipe, pip, line, touch, clip, modulator, servo, dsc, booster, underbody, foul, rhd, hose, vacuum, distance, rout, grommet, stud, flexi
		186	brake, dsc, caliper, park, pad, emf, pedal, fluid, disc, booster, actuator, vacuum, unit, switch, master, betroffene, optional, repair, lligkeit, rotor
		248	handbrake, grip, lever, cable, ratchet, click, equaliser, gaiter, caliper, gaitor, brake, hold, button, probertplease, retain, lug, greenhandshaken, release, rft, pawl
	seat	3	seat, cushion, backrest, leather, crease, wavy, bolster, wrinkle, headrest, isofix, allocation, rear, stitch, thigh, cover, squab, audit, seam, material, lh
		6	seat, noise, creak, backrest, armrest, squeak, headrest, adjust, lean, rest, click, hear, adjustment, cushion, forward, driver, weight, passenger, arm, static
		8	belt, seatbelt, buckle, seat, retract, retractor, safety, twist, receiver, row, trap, web, rear, pillar, pull, anchor, passenger, loop, middle, reel
		13	seat, headrest, backrest, lumbar, lever, adjustment, easy, function, switch, release, fold, lfe, forward, adjust, bowden, row, passenger, inflate, rep, rest
		50	heat, seat, heater, fslesen, lin, button, switch, hex, fortnr, seatmoduledriver, circuit, function, job, driver, erg, backrest, mat, passenger, fehlertext, sgbd
		72	rattle, belt, seat, retractor, headrest, seatbelt, noise, cobble, cobblestone, stone, surface, reel, buckle, passenger, drive, faurecia, hear, road, row, buzz
	mirror	10	mirror, exterior, view, fold, switch, outside, door, triangle, rearview, glass, base, cap, function, leave, right, adjust, interior, driver, adjustment, heat
		110	mirror, exterior, fold, noise, view, vibration, vibrate, squeak, rearview, rattle, door, close, hear, object, outside, interior, creak, surface, roll, disturb
		122	visor, vanity, sun, sunvisor, mirror, light, inoperative, makeup, illumination, flicker, stay, vdc, st, function, intermittent, grant, illuminate, turn, close, akhtar
		247	mirror, whistle, wind, cap, bezel, triangle, noise, exterior, tissiman, atc, km, windnoise, foam, hear, outer, rush, come, overall, disturb, start

Table A.6.: Per each number of topic clusters, topic word examples generated by BERTopic model, for PQM data, range 50-250

Number of topic clusters	Keyword	Topic ID	Topic words
80	brake	25	brake, pedal, fluid, pipe, caliper, pad, dsc, vacuum, booster, master, accelerator, line, servo, pip, air, vehicle, clip, disc, leak, customer
		66	brake, pad, noise, park, disc, pedal, squeal, squeak, wear, caliper, speed, sensor, emf, rotor, hear, apply, drive, release, car, rear
	seat	0	seat, backrest, leather, cushion, headrest, rear, armrest, crease, bolster, wavy, cover, rest, audit, leave, wrinkle, gap, squab, issue, row, visible
		7	belt, seat, seatbelt, buckle, retract, retractor, rattle, rear, safety, row, receiver, pillar, twist, trap, passenger, pull, right, issue, leave, noise
		13	seat, noise, backrest, squeak, creak, armrest, headrest, adjust, click, adjustment, driver, rest, hear, rear, lean, cushion, forward, passenger, leather, weight
	mirror	10	mirror, exterior, view, fold, door, cap, switch, glass, outside, triangle, leave, right, base, rearview, function, noise, driver, rear, cover, interior
85	brake	40	brake, pedal, clutch, fluid, pad, caliper, dsc, master, accelerator, booster, vacuum, cylinder, leak, wear, vehicle, servo, replace, customer, bleed, pipe
		60	brake, pad, park, noise, disc, pedal, squeal, wear, squeak, sale, caliper, sensor, rotor, speed, emf, vehicle, pam, hear, apply, inform
		81	pipe, brake, air, pip, clip, line, touch, intake, fit, dsc, box, servo, filter, cap, aircon, snorkel, modulator, cleaner, duct, clamp
	seat	3	seat, noise, backrest, armrest, creak, headrest, squeak, adjust, adjustment, rest, lumbar, driver, forward, function, rear, passenger, hear, position, arm, click
		7	belt, seat, seatbelt, buckle, retract, retractor, rattle, rear, safety, row, twist, receiver, pillar, passenger, trap, pull, right, leave, fit, find
		10	audit, wavy, qzs, seat, qz, dashboard, backrest, picture, gap, find, rh, rear, lh, bow, issue, allocate, leave, middle, tension, right
	mirror	34	heat, hd, seat, fslesen, hex, test, fortnr, job, fehlertext, error, sgbd, temperature, pressure, cycle, erg, ist, run, fail, disassembly, disturbance
		47	mirror, exterior, fold, view, switch, door, function, adjust, outside, leave, right, adjustment, noise, glass, driver, rear, heat, lin, road, interior
		75	mirror, cap, exterior, triangle, base, whistle, rearview, seal, door, cover, view, outer, wind, noise, fit, gap, leave, audit, bezel, window
90	brake	29	brake, pedal, fluid, clutch, caliper, pad, accelerator, master, booster, dsc, vacuum, leak, cylinder, park, servo, wear, replace, customer, vehicle, bleed
		67	brake, pad, handbrake, noise, disc, pedal, squeal, wear, squeak, caliper, speed, sensor, rotor, hear, apply, cable, drive, rear, axle, stop
		89	pipe, brake, pip, clip, air, line, touch, intake, filter, fit, servo, box, suction, jubilee, dsc, vacuum, duct, secure, turbo, snorkel
	seat	0	seat, leather, backrest, cushion, headrest, rear, armrest, crease, bolster, cover, wavy, rest, audit, wrinkle, leave, squab, row, issue, visible, foam
		7	belt, seat, seatbelt, buckle, retract, retractor, rattle, rear, safety, row, passenger, pillar, twist, receiver, pull, trap, right, noise, issue, reel
		9	seat, noise, creak, backrest, armrest, squeak, headrest, adjust, hear, static, click, rest, driver, lean, rear, adjustment, cushion, forward, arm, passenger
	mirror	38	heat, hd, seat, test, shield, cycle, temperature, disassembly, disturbance, run, ptce, damage, htoe, rotor, heatshield, lin, detect, circuit, heater, analysis
		10	mirror, exterior, view, fold, door, switch, glass, cap, outside, triangle, leave, base, right, rearview, function, driver, rear, cover, noise, interior
		74	visor, sun, sunvisor, vanity, mirror, light, grab, driver, vdc, fit, clip, hook, headliner, inoperative, cover, handle, passenger, issue, loose, function
95	brake	19	brake, pedal, clutch, fluid, caliper, pad, pipe, master, booster, dsc, vacuum, accelerator, line, vehicle, replace, perform, breakdown, cylinder, leak, repair
		77	brake, pad, noise, park, disc, pedal, squeal, wear, squeak, caliper, rotor, speed, emf, sensor, hear, apply, drive, release, switch, car
	seat	0	seat, backrest, leather, cushion, headrest, rear, armrest, crease, bolster, wavy, cover, rest, audit, wrinkle, row, leave, stitch, visible, gap, issue
		5	belt, seat, seatbelt, buckle, retract, retractor, rear, rattle, safety, row, receiver, twist, pillar, passenger, trap, pull, right, web, reel, issue
		13	seat, noise, backrest, creak, armrest, squeak, vfc, headrest, adjust, driver, static, click, rest, lean, hear, rear, mingxiang, forward, adjustment, cushion
		51	heat, hd, seat, test, cycle, temperature, disassembly, disturbance, ptce, run, damage, rotor, htoe, circuit, detect, lin, heater, analysis, machine, mech
	mirror	10	mirror, exterior, view, fold, door, cap, switch, glass, outside, rearview, leave, triangle, base, right, function, driver, noise, interior, adjust, rear
		85	visor, sun, sunvisor, vanity, mirror, light, grab, vdc, hook, fit, headliner, driver, clip, inoperative, handle, cover, issue, loose, passenger, pivot

Table A.7.: Per each number of topic clusters, topic word examples generated by BERTopic model, for PQM data, range 80-95

List of Figures

List of Tables

Acronyms

BERT Bidirectional Encoder Representations from Transformers. 11, 12, 14

BI Business Intelligence. 77

BMW Bayerische MotorWerke. v, 1, 2, 4, 18, 20, 38, 59, 70, 76–78, 91

BoW Bag-of-words. 8, 9, 13, 14

CBOW Continuous Bag-of-Words. 9

CTM Correlated Topic Model. 13

DBSCAN Density-Based Spatial Clustering of Applications with Noise. 12

ELMO Embeddings from Language Model. 11, 14

EM Expectation-Maximization. 14

GPT Generative Pre-trained Transformer. 11, 14

HDBSCAN Hierarchical Density-Based Spatial Clustering of Applications with Noise. 12, 14, 28

LDA Latent Dirichlet Allocation. 3, 5, 13–15, 17, 27, 28, 32, 35, 37, 40–45, 48, 50, 51, 76–78, 91

LSA Latent Semantic Analysis. 3, 10, 13

LSI Latent Semantic Indexing. 13, 15

MLM Masked Language Model. 11

NLP Natural Language Processing. v, 1, 3–7, 11, 15, 29, 60, 67, 77

NLTK Natural Language Toolkit. 25

NMF Non-negative Matrix Factorization. 13

NSP Next Sentence Prediction. 11, 12

OOV out-of-vocabulary. 10, 11

PCA Principal Component Analysis. 14

pLSA Probabilistic Latent Semantic Analysis. 13

PQM Problem Quality Management. 2, 3, 19, 20, 22, 25–28, 32, 35–37, 39, 40, 46–48, 51–53, 55, 59–62, 64, 66, 71, 78, 87, 91

RoBERTa Robustly Optimized BERT Pretraining Approach. 12, 14

SVD Singular Value Decomposition. 13

TF-IDF Term Frequency - Inverse Document Frequency. 8, 9, 13, 15, 28, 29, 52

t-SNE t-Distributed Stochastic Neighbor Embedding. 14

UMAP Uniform Manifold Approximation and Projection for Dimension Reduction. 14, 28, 77

Bibliography

[1] R. Khoshkangini, P. Sheikholharam Mashhadi, P. Berck, S. Gholami Shahbandi, S. Pashami, S. Nowaczyk, and T. Niklasson. "Early Prediction of Quality Issues in Automotive Modern Industry". In: *Information* 11.7 (2020). ISSN: 2078-2489. DOI: 10.3390/info11070354. URL: https://www.mdpi.com/2078-2489/11/7/354.

[2] D. M. Blei, A. Y. Ng, and M. I. Jordan. "Latent Dirichlet Allocation". In: *J. Mach. Learn. Res.* 3.null (Mar. 2003), pp. 993–1022. ISSN: 1532-4435. DOI: 10.5555/944919.944937.

[3] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. "Indexing by Latent Semantic Analysis". In: *J. Am. Soc. Inf. Sci.* 41.6 (1990), pp. 391–407. DOI: https://doi.org/10.1002/(SICI)1097-4571(199009)41:6<391::AID-ASI1>3.0.CO;2-9. URL: https://doi.org/10.1002/(SICI)1097-4571(199009)41:6%3C391::AID-ASI1%3E3.0.CO;2-9.

[4] T. Mikolov, W.-t. Yih, and G. Zweig. "Linguistic Regularities in Continuous Space Word Representations". In: *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Atlanta, Georgia: Association for Computational Linguistics, June 2013, pp. 746–751. URL: https://aclanthology.org/N13-1090.

[5] D. Angelov. *Top2Vec: Distributed Representations of Topics*. 2020. DOI: 10.48550/ARXIV.2008.09470. URL: https://arxiv.org/abs/2008.09470.

[6] D. Chandrasekaran and V. Mago. "Evolution of Semantic Similarity—A Survey". In: *ACM Computing Surveys* 54.2 (Mar. 2022), pp. 1–37. DOI: 10.1145/3440755. URL: https://doi.org/10.1145/3440755.

[7] G. Sidorov, A. Gelbukh, H. Gomez Adorno, and D. Pinto. "Soft Similarity and Soft Cosine Measure: Similarity of Features in Vector Space Model". In: *Computación y Sistemas* 18 (Sept. 2014). DOI: 10.13053/cys-18-3-2043.

[8] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *arXiv preprint arXiv:1810.04805* (2018).

[9] A. Agnihotri. *Presenting machine learning model results as Business Insights*. June 2021. URL: https://towardsdatascience.com/presenting-machine-learning-model-results-as-business-insights-b42309c5e974.

[10] M. Grootendorst. *BERTopic: Neural topic modeling with a class-based TF-IDF procedure*. 2022. DOI: 10.48550/ARXIV.2203.05794. URL: https://arxiv.org/abs/2203.05794.

[11] M. Grinberg. *Flask web development: developing web applications with Python*. O'Reilly Media, Inc., 2018.

[12] S. Kang and S. Kim. "Lessons Learned from Topic Modeling Analysis of COVID-19 News to Enrich Statistics Education in Korea". In: *Sustainability* 14.6 (Mar. 2022), p. 3240. ISSN: 2071-1050. DOI: 10.3390/su14063240. URL: http://dx.doi.org/10.3390/su14063240.

[13] *Structured vs. Unstructured Data: A complete guide*. URL: https://www.talend.com/resources/structured-vs-unstructured-data/.

[14] 2. Bill Inmon. *Why do we call text "Unstructured"?* June 2016. URL: https://tdwi.org/articles/2016/06/28/text-unstructured.aspx.

[15] D. Jurafsky and J. H. Martin. *Speech and language processing : an introduction to natural language processing, computational linguistics, and speech recognition*. Upper Saddle River, N.J.: Pearson Prentice Hall, 2009. ISBN: 9780131873216 0131873210. URL: http://www.amazon.com/Speech-Language-Processing-2nd-Edition/dp/0131873210/ref=pd_bxgy_b_img_y.

[16] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Web publication at informationretrieval.org. Cambridge University Press, 2008.

[17] J. Pennington, R. Socher, and C. D. Manning. "Glove: Global Vectors for Word Representation." In: *EMNLP*. Vol. 14. 2014, pp. 1532–1543.

[18] T. Mikolov, K. Chen, G. Corrado, and J. Dean. *Efficient Estimation of Word Representations in Vector Space*. 2013. DOI: 10.48550/ARXIV.1301.3781. URL: https://arxiv.org/abs/1301.3781.

[19] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. *Distributed Representations of Words and Phrases and their Compositionality*. 2013. DOI: 10.48550/ARXIV.1310.4546. URL: https://arxiv.org/abs/1310.4546.

[20] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov. "Enriching Word Vectors with Subword Information". In: *Transactions of the Association for Computational Linguistics* 5 (2017), pp. 135–146. ISSN: 2307-387X.

[21] F. Almeida and G. Xexéo. *Word Embeddings: A Survey*. 2019. DOI: 10.48550/ARXIV.1901.09069. URL: https://arxiv.org/abs/1901.09069.

[22] A. Torfi, R. A. Shirvani, Y. Keneshloo, N. Tavaf, and E. A. Fox. *Natural Language Processing Advancements By Deep Learning: A Survey*. 2020. DOI: 10.48550/ARXIV.2003.01200. URL: https://arxiv.org/abs/2003.01200.

[23] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov. "Bag of Tricks for Efficient Text Classification". In: *arXiv preprint arXiv:1607.01759* (2016).

[24] A. Joulin, E. Grave, P. Bojanowski, M. Douze, H. Jégou, and T. Mikolov. "FastText.zip: Compressing text classification models". In: *arXiv preprint arXiv:1612.03651* (2016).

[25] Q. Liu, M. J. Kusner, and P. Blunsom. *A Survey on Contextual Embeddings*. 2020. DOI: 10.48550/ARXIV.2003.07278. URL: https://arxiv.org/abs/2003.07278.

[26] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer. *Deep contextualized word representations*. 2018. DOI: 10.48550/ARXIV.1802.05365. URL: https://arxiv.org/abs/1802.05365.

[27] V. Cohen and A. Gokaslan. "OpenGPT-2: Open Language Models and Implications of Generated Text". In: *XRDS* 27.1 (Sept. 2020), pp. 26–30. ISSN: 1528-4972. DOI: 10.1145/3416063. URL: https://doi.org/10.1145/3416063.

[28] A. Miaschi and F. Dell'Orletta. "Contextual and Non-Contextual Word Embeddings: an in-depth Linguistic Investigation". In: *Proceedings of the 5th Workshop on Representation Learning for NLP*. Online: Association for Computational Linguistics, July 2020, pp. 110–119. DOI: 10.18653/v1/2020.repl4nlp-1.15. URL: https://aclanthology.org/2020.repl4nlp-1.15.

[29] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. *Attention Is All You Need*. 2017. DOI: 10.48550/ARXIV.1706.03762. URL: https://arxiv.org/abs/1706.03762.

[30] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. 2019. DOI: 10.48550/ARXIV.1907.11692. URL: https://arxiv.org/abs/1907.11692.

[31] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut. *ALBERT: A Lite BERT for Self-supervised Learning of Language Representations*. 2019. DOI: 10.48550/ARXIV.1909.11942. URL: https://arxiv.org/abs/1909.11942.

[32] N. Reimers and I. Gurevych. "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Nov. 2019. URL: https://arxiv.org/abs/1908.10084.

[33] P. Rai and S. Shubha. "A Survey of Clustering Techniques". In: *International Journal of Computer Applications* 7 (Oct. 2010). DOI: 10.5120/1326-1808.

[34] S. Lloyd. "Least squares quantization in PCM". In: *IEEE Transactions on Information Theory* 28.2 (1982), pp. 129–137. DOI: 10.1109/TIT.1982.1056489.

[35] S. Bano and N. Khan. "A Survey of Data Clustering Methods". In: *International Journal of Advanced Science and Technology* 113 (Apr. 2018). DOI: 10.14257/ijast.2018.113.14.

[36] R. J. G. B. Campello, D. Moulavi, and J. Sander. "Density-Based Clustering Based on Hierarchical Density Estimates". In: *Advances in Knowledge Discovery and Data Mining*. Ed. by J. Pei, V. S. Tseng, L. Cao, H. Motoda, and G. Xu. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 160–172. ISBN: 978-3-642-37456-2.

[37] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise". In: *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*. KDD'96. Portland, Oregon: AAAI Press, 1996, pp. 226–231.

[38] L. McInnes and J. Healy. "Accelerated Hierarchical Density Based Clustering". In: *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*. IEEE, Nov. 2017. DOI: 10.1109/icdmw.2017.12. URL: https://doi.org/10.1109%2Ficdmw.2017.12.

[39] J. Boyd-Graber, Y. Hu, and D. Minmo. *Applications of Topic Models*. 2017. DOI: 10.1561/1500000030.

[40] D. Blei and J. Lafferty. "Correlated Topic Models". In: *ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS* 18 (2006), p. 147. URL: http://www.cs.cmu.edu/~lafferty/pub/ctm.pdf.

[41] R. Alghamdi and K. Alfalqi. "A Survey of Topic Modeling in Text Mining". In: *International Journal of Advanced Computer Science and Applications* 6.1 (2015). DOI: 10.14569/IJACSA.2015.060121. URL: http://dx.doi.org/10.14569/IJACSA.2015.060121.

[42] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval*. USA: McGraw-Hill, Inc., 1986. ISBN: 0070544840.

[43] T. Hofmann. "Probabilistic Latent Semantic Indexing". In: *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '99. Berkeley, California, USA: Association for Computing Machinery, 1999, pp. 50–57. ISBN: 1581130961. DOI: 10.1145/312624.312649. URL: https://doi.org/10.1145/312624.312649.

[44] P. Kherwa and P. Bansal. "Topic Modeling: A Comprehensive Review". In: *EAI Endorsed Transactions on Scalable Information Systems* 7.24 (July 2019). DOI: 10.4108/eai.13-7-2018.159623.

[45] Y. Jiang, J. Liu, Z. Li, P. Li, and H. Lu. "Co-regularized PLSA for Multi-view Clustering". In: *Computer Vision – ACCV 2012*. Ed. by K. M. Lee, Y. Matsushita, J. M. Rehg, and Z. Hu. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 202–213. ISBN: 978-3-642-37444-9.

[46] C. Boye Asmussen and C. Møller. "Smart literature review: a practical topic modelling approach to exploratory literature review". In: *Journal of Big Data* 6 (Oct. 2019). DOI: 10.1186/s40537-019-0255-7.

[47] P. Ma, Q. Zeng-Treitler, and S. J. Nelson. "Use of two topic modeling methods to investigate COVID vaccine hesitancy". In: *Proceedings 14th International Conference on ICT, Society and Human Beings (ICT 2021), the 18th International Conference Web Based Communities and Social Media (WBC 2021)* (2021). DOI: 10.33965/eh2021_202106c030.

[48] R. Egger and J. Yu. "A Topic Modeling Comparison Between LDA, NMF, Top2Vec, and BERTopic to Demystify Twitter Posts". In: *Frontiers in Sociology* 7 (2022). ISSN: 2297-7775. DOI: 10.3389/fsoc.2022.886498. URL: https://www.frontiersin.org/articles/10.3389/fsoc.2022.886498.

[49] D. Hendry, F. Darari, R. Nurfadillah, G. Khanna, M. Sun, P. C. Condylis, and N. Taufik. "Topic Modeling for Customer Service Chats". In: *2021 International Conference on Advanced Computer Science and Information Systems (ICACSIS)*. 2021, pp. 1–6. DOI: 10.1109/ICACSIS53237.2021.9631322.

[50] F. Nan, R. Ding, R. Nallapati, and B. Xiang. *Topic Modeling with Wasserstein Autoencoders*. 2019. DOI: 10.48550/ARXIV.1907.12374. URL: https://arxiv.org/abs/1907.12374.

[51] M. Smatana and P. Butka. "TopicAE: A Topic Modeling Autoencoder". In: *Acta Polytechnica Hungarica* 16.4 (2019). DOI: 10.12700/aph.16.4.2019.4.4.

[52] A. B. Dieng, C. Wang, J. Gao, and J. Paisley. *TopicRNN: A Recurrent Neural Network with Long-Range Semantic Dependency*. 2016. DOI: 10.48550/ARXIV.1611.01702. URL: https://arxiv.org/abs/1611.01702.

[53] Y. Meng, Y. Zhang, J. Huang, Y. Zhang, and J. Han. "Topic Discovery via Latent Space Clustering of Pretrained Language Model Representations". In: *Proceedings of the ACM Web Conference 2022*. ACM, Apr. 2022. DOI: 10.1145/3485447.3512034.

[54] H. Zhao, D. Phung, V. Huynh, Y. Jin, L. Du, and W. Buntine. *Topic Modelling Meets Deep Neural Networks: A Survey*. 2021. DOI: 10.48550/ARXIV.2103.00498. URL: https://arxiv.org/abs/2103.00498.

[55] Y. Chaudhary, P. Gupta, K. Saxena, V. Kulkarni, T. Runkler, and H. Schütze. *TopicBERT for Energy Efficient Document Classification*. 2020. DOI: 10.48550/ARXIV.2010.16407. URL: https://arxiv.org/abs/2010.16407.

[56] A. Hoyle, P. Goel, and P. Resnik. *Improving Neural Topic Models using Knowledge Distillation*. 2020. DOI: 10.48550/ARXIV.2010.02377. URL: https://arxiv.org/abs/2010.02377.

[57] F. Bianchi, S. Terragni, and D. Hovy. *Pre-training is a Hot Topic: Contextualized Document Embeddings Improve Topic Coherence*. 2020. DOI: 10.48550/ARXIV.2004.03974. URL: https://arxiv.org/abs/2004.03974.

[58] L. Thompson and D. Mimno. *Topic Modeling with Contextualized Word Representation Clusters*. 2020. DOI: 10.48550/ARXIV.2010.12626. URL: https://arxiv.org/abs/2010.12626.

[59] S. Sia, A. Dalmia, and S. J. Mielke. *Tired of Topic Models? Clusters of Pretrained Word Embeddings Make for Fast and Good Topics too!* 2020. DOI: 10.48550/ARXIV.2004.14914. URL: https://arxiv.org/abs/2004.14914.

[60] Q. V. Le and T. Mikolov. *Distributed Representations of Sentences and Documents*. 2014. DOI: 10.48550/ARXIV.1405.4053. URL: https://arxiv.org/abs/1405.4053.

[61] L. McInnes, J. Healy, and J. Melville. *UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction*. 2018. DOI: 10.48550/ARXIV.1802.03426. URL: https://arxiv.org/abs/1802.03426.

[62] G. A. Miller. "WordNet: A Lexical Database for English". In: *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*. 1994. URL: https://aclanthology.org/H94-1111.

[63] S. Al-Anazi, H. AlMahmoud, and I. Al-Turaiki. "Finding Similar Documents Using Different Clustering Techniques". In: *Procedia Computer Science* 82 (2016). 4th Symposium on Data Mining Applications, SDMA2016, 30 March 2016, Riyadh, Saudi Arabia, pp. 28–34. ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2016.04.005. URL: https://www.sciencedirect.com/science/article/pii/S1877050916300199.

[64] D. Charlet and G. Damnati. "SimBow at SemEval-2017 Task 3: Soft-Cosine Semantic Similarity between Questions for Community Question Answering". In: *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Vancouver, Canada: Association for Computational Linguistics, Aug. 2017, pp. 315–319. DOI: 10.18653/v1/S17-2051. URL: https://aclanthology.org/S17-2051.

[65] L. Peter, I. Chakraborty, and S. Banerjee. "AI applications to customer feedback research: A review". English. In: *Review of Marketing Research*. Ed. by K. Sudhir and O. Toubia. Emerald Publishing, June 2022.

[66] D. Proserpio, J. Hauser, X. Liu, T. Amano, A. Burnap, T. Guo, D. Lee, R. Lewis, K. Misra, E. Schwarz, A. Timoshenko, L. Xu, and H. Yoganarasimhan. "Soul and machine (learning)". In: *Marketing Letters* 31 (Dec. 2020). DOI: 10.1007/s11002-020-09538-4.

[67] J. Salminen, C. Kandpal, A. M. Kamel, S.-g. Jung, and B. J. Jansen. "Creating and detecting fake reviews of online products". In: *Journal of Retailing and Consumer Services* 64 (2022), p. 102771. ISSN: 0969-6989. DOI: https://doi.org/10.1016/j.jretconser.2021.102771. URL: https://www.sciencedirect.com/science/article/pii/S0969698921003374.

[68] R. Jain and N. Mital. "ANALYZING CUSTOMER FEEDBACK FOR IMPROVED SERVICE QUALITY USING BINARY LOGISTICS REGRESSION MODEL". In: *Administrative Development 'A Journal of HIPA, Shimla'* 8 (Oct. 2021), p. 2021. DOI: 10.53338/ADHIPA2021.V08.Si01.11.

[69] K. L. S. Kumar, J. Desai, and J. Majumdar. "Opinion mining and sentiment analysis on online customer review". In: *2016 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)*. 2016, pp. 1–4. DOI: 10.1109/ICCIC.2016.7919584.

[70] D. Das, S. Sharma, S. Natani, N. Khare, and B. Singh. "Sentimental Analysis for Airline Twitter data". In: *IOP Conference Series: Materials Science and Engineering* 263 (Nov. 2017), p. 042067. DOI: 10.1088/1757-899X/263/4/042067.

[71] J. Jotheeswaran. "Opinion mining using decision tree based feature selection through Manhattan hierarchical cluster measure". In: *Journal of Theoretical and Applied Information Technology* 58 (Dec. 2013), pp. 72–80.

[72] Imamah, Husni, E. M. Rachman, I. O. Suzanti, and F. A. Mufarroha. "Text Mining and Support Vector Machine for Sentiment Analysis of Tourist Reviews in Bangkalan Regency". In: *Journal of Physics: Conference Series* 1477.2 (Mar. 2020), p. 022023. DOI: 10.1088/1742-6596/1477/2/022023. URL: https://doi.org/10.1088/1742-6596/1477/2/022023.

[73] G. Singh and R. Bhatia. "A KNN BASED TECHNIQUE ON OPINION MINING USING SEMANTIC ANALYSIS". In: (June 2020). DOI: 10.24941/ijcr.32244.09.2018.

[74] M. Wankhade, A. Chandra, S. Rao, S. Dara, and B. Kaushik. "A Sentiment Analysis of Food Review using Logistic Regression". In: (Sept. 2017), pp. 2456–3307.

[75] S. F. Eletter, K. I. AlQeisi, and G. A. Elrefae. "The Use of Topic Modeling in Mining Customers' Reviews". In: *2021 22nd International Arab Conference on Information Technology (ACIT)*. 2021, pp. 1–4. DOI: 10.1109/ACIT53391.2021.9677049.

[76] S. Tirunillai and G. J. Tellis. "Mining Marketing Meaning from Online Chatter: Strategic Brand Analysis of Big Data Using Latent Dirichlet Allocation". In: *Journal of Marketing Research* 51.4 (2014), pp. 463–479. DOI: 10.1509/jmr.12.0106. eprint: https://doi.org/10.1509/jmr.12.0106. URL: https://doi.org/10.1509/jmr.12.0106.

[77] M. Honnibal and I. Montani. "spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing". To appear. 2017.

[78] E. Loper and S. Bird. "NLTK: The Natural Language Toolkit". In: *CoRR* cs.CL/0205028 (2002). URL: http://dblp.uni-trier.de/db/journals/corr/corr0205.html#cs-CL-0205028.

[79] A. Hoyle, P. Goel, D. Peskov, A. Hian-Cheong, J. Boyd-Graber, and P. Resnik. *Is Automated Topic Model Evaluation Broken?: The Incoherence of Coherence*. 2021. DOI: 10.48550/ARXIV.2107.02173. URL: https://arxiv.org/abs/2107.02173.

[80] J. Chang, J. Boyd-Graber, S. Gerrish, C. Wang, and D. Blei. "Reading Tea Leaves: How Humans Interpret Topic Models". In: vol. 32. Jan. 2009, pp. 288–296.

[81] D. Mimno, H. M. Wallach, E. Talley, M. Leenders, and A. McCallum. "Optimizing Semantic Coherence in Topic Models". In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. EMNLP '11. Edinburgh, United Kingdom: Association for Computational Linguistics, 2011, pp. 262–272. ISBN: 9781937284114.

[82] M. Röder, A. Both, and A. Hinneburg. "Exploring the Space of Topic Coherence Measures". In: *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*. WSDM '15. Shanghai, China: Association for Computing Machinery, 2015, pp. 399–408. ISBN: 9781450333177. DOI: 10.1145/2684822.2685324. URL: https://doi.org/10.1145/2684822.2685324.

[83] D. Newman, J. H. Lau, K. Grieser, and T. Baldwin. "Automatic Evaluation of Topic Coherence". In: *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. HLT '10. Los Angeles, California: Association for Computational Linguistics, 2010, pp. 100–108. ISBN: 1932432655.

[84] G. Bouma. "Normalized (pointwise) mutual information in collocation extraction". In: 2009.

[85] K. Stevens, W. P. Kegelmeyer, D. Andrzejewski, and D. Buttler. "Exploring Topic Coherence over Many Models and Many Topics." In: *EMNLP-CoNLL*. Ed. by J. Tsujii, J. Henderson, and M. Pasca. ACL, 2012, pp. 952–961. ISBN: 978-1-937284-43-5. URL: http://dblp.uni-trier.de/db/conf/emnlp/emnlp2012.html#StevensKAB12.

[86] R. Atenstaedt. *Word cloud analysis of the BJGP*. Mar. 2012. URL: https://doi.org/10.3399/bjgp12X630142.

[87] V. U. Thompson, C. Panchev, and M. Oakes. "Performance evaluation of similarity measures on similar and dissimilar text retrieval". In: *2015 7th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K)*. Vol. 01. 2015, pp. 577–584.

[88] P. Achananuparp, X. Hu, and X. Shen. "The Evaluation of Sentence Similarity Measures". In: vol. 5182. Sept. 2008, pp. 305–316. ISBN: 978-3-540-85835-5. DOI: 10.1007/978-3-540-85836-2_29.

[89] R. Rehurek and P. Sojka. "Gensim–python framework for vector space modelling". In: *NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic* 3.2 (2011).

[90] C. Sievert and K. Shirley. "LDAvis: A method for visualizing and interpreting topics". In: *Proceedings of the Workshop on Interactive Language Learning, Visualization, and Interfaces*. Baltimore, Maryland, USA: Association for Computational Linguistics, June 2014, pp. 63–70. DOI: 10.3115/v1/W14-3110. URL: https://aclanthology.org/W14-3110.

[91] P. T. Inc. *Collaborative data science*. 2015. URL: https://plot.ly.

[92] 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. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[93] V. Novotný. "Implementation Notes for the Soft Cosine Measure". In: *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. CIKM '18. Torino, Italy: Association for Computing Machinery, 2018, pp. 1639–1642. ISBN: 9781450360142. DOI: 10.1145/3269206.3269317. URL: https://doi.org/10.1145/3269206.3269317.

[94] L. Galke, A. Saleh, and A. Scherp. "Evaluating the Impact of Word Embeddings on Similarity Scoring in Practical Information Retrieval". In: Zenodo, Sept. 2017. DOI: 10.5281/zenodo.1143963. URL: https://doi.org/10.5281/zenodo.1143963.

[95] C. Wang, P. Nulty, and D. Lillis. "A Comparative Study on Word Embeddings in Deep Learning for Text Classification". In: Dec. 2020, pp. 37–46. DOI: 10.1145/3443279.3443304.

[96] D. Dessí, R. Helaoui, V. Kumar, D. R. Recupero, and D. Riboni. "TF-IDF vs Word Embeddings for Morbidity Identification in Clinical Notes: An Initial Study". In: (2020). DOI: 10.5281/ZENODO.4777594. URL: https://zenodo.org/record/4777594.

[97] *Bootstrap. The most popular HTML, CSS, and JS library in the world.* `https://getbootstrap.com`. Accessed: 2022-07-30.

[98] *DataTables | Table plug-in for JQuery*. `https://datatables.net`. Accessed: 2022-07-30.