


Article

From Data to Human-Readable Requirements: Advancing Requirements Elicitation through Language-Transformer-Enhanced Opportunity Mining

Pascal Harth ^{1,†}, Orlando Jähde ^{1,†} , Sophia Schneider ^{1,†} , Nils Horn ^{1,2}  and Rüdiger Buchkremer ^{1,*} 

¹ Institute of IT Management and Digitization Research (IFID), 40476 Düsseldorf, Germany; pascal.harth@fom-net.de (P.H.); orlando.jaehde@fom-net.de (O.J.); sophia.schneider@fom-net.de (S.S.); nhorn@alu.ucam.edu (N.H.)

² Faculty of Legal and Business Sciences, Universidad Católica San Antonio de Murcia, 30107 Murcia, Spain

* Correspondence: ruediger.buchkremer@fom.de

† These authors contributed equally to this work.

Abstract: In this research, we present an algorithm that leverages language-transformer technologies to automate the generation of product requirements, utilizing E-Shop consumer reviews as a data source. Our methodology combines classical natural language processing techniques with diverse functions derived from transformer concepts, including keyword and summary generation. To effectively capture the most critical requirements, we employ the opportunity matrix as a robust mechanism for identifying and prioritizing urgent needs. Utilizing transformer technologies, mainly through the implementation of summarization and sentiment analysis, we can extract fundamental requirements from consumer assessments. As a practical demonstration, we apply our technology to analyze the ratings of the Amazon echo dot, showcasing our algorithm's superiority over conventional approaches by extracting human-readable problem descriptions to identify critical user needs. The results of our study exemplify the potential of transformer-enhanced opportunity mining in advancing the requirements-elicitation processes. Our approach streamlines product improvement by extracting human-readable problem descriptions from E-Shop consumer reviews, augmenting operational efficiency, and facilitating decision-making. These findings underscore the transformative impact of incorporating transformer technologies within requirements engineering, paving the way for more effective and scalable algorithms to elicit and address user needs.

Keywords: requirements engineering; requirements elicitation; consumer reviews; online reviews; deep learning; language transformers; machine learning; natural language processing



Citation: Harth, P.; Jähde, O.; Schneider, S.; Horn, N.; Buchkremer, R. From Data to Human-Readable Requirements: Advancing Requirements Elicitation through Language-Transformer-Enhanced Opportunity Mining. *Algorithms* **2023**, *16*, 403. <https://doi.org/10.3390/a16090403>

Academic Editor: Frank Werner

Received: 13 July 2023

Revised: 17 August 2023

Accepted: 22 August 2023

Published: 23 August 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

In recent years, there has been a convergence between the fields of requirements-engineering and semantic technologies [1]. The progress in semantic concepts and technologies opens up new possibilities for utilizing natural language processing, artificial intelligence, and language-transformer technologies to define information and knowledge semantics across various domains.

The Internet has become a powerful platform for online shopping, combined with interpersonal communication, leading consumers to increasingly rely on online sources for information about products they are interested in purchasing. Online reviews have experienced explosive growth in recent years, with the most prominent E-commerce platforms hosting hundreds of millions of reviews. The abundance of product feedback conveyed through online reviews has presented firms with a compelling new source of information. When revising their products, designers often consider multiple data sources, such as focus groups, consumer complaints, warranty claims, and ingenuity. Experimental evidence suggests that incorporating consumer feedback into product design, instead of solely relying

on laboratory-based approaches, leads to improved sales performance [2]. Consequently, there is a growing call in academia and industry for methods that prioritize consumer-driven innovation, as addressing consumer preferences has enhanced satisfaction and competitive positioning.

However, the sheer volume of online reviews makes it nearly impossible for practitioners to analyze them comprehensively and manually. Therefore, automated methods for analyzing the content of online reviews offer the potential for significant time savings and ensure that firms can process and respond to critical feedback efficiently. Integrating these methods into the requirements-elicitation process holds great promise for transforming large amounts of data into human-readable requirements, enabling a deeper understanding of consumer preferences, driving processes and product innovation.

To address this gap, we propose a requirements-elicitation algorithm based on domain-specific natural language processing or language-transformer technologies. Within requirements elicitation, “domain-specific” embodies a focused and finely-tuned approach that harmonizes seamlessly with a specific domain’s distinctive attributes, terminologies, and intricacies. This process entails the meticulous tailoring of our algorithmic framework to adjust effortlessly to the nuances and particulars of the Amazon Echo Dot domain. Our algorithm surpasses traditional known approaches in enhancing the quality of the process and product innovation derived from online reviews. Lim et al. review the current state-of-the-art procedures for data-driven requirements elicitation from dynamic data sources. The findings indicate that existing automated requirements elicitation predominantly relies on human-sourced data, particularly online reviews, as requirements sources and employs supervised machine-learning techniques for data processing. However, these approaches often result in merely identifying and classifying requirements-related information or features without generating requirements in a readily usable form. Their article highlights the pressing need to develop methods to leverage process-mediated and machine-generated data in requirements elicitation [3].

Overall, our paper aims to advance requirements engineering by exploring the potential of an algorithm via language-transformer-enhanced opportunity mining from online reviews, empowered by natural language processing, artificial intelligence, or language-transformer technologies, to bridge the gap between data and human-readable requirements.

2. Background and Related Work

2.1. Requirements Elicitation

The requirements-elicitation process is considered one of the main activities in requirements engineering [4]. The primary goal of the activity is to elicit new requirements. Traditionally, various techniques, including conducting surveys, observations, or interviews and workshops, are used by a requirements engineer to gather new requirements [5]. This activity often utilizes structured requirements templates that follow pre-defined notations. The requirements-elicitation process generally includes various steps that, according to Zowghi and Coulin [6], can be divided into five fundamental activities: understanding the application domain—where the environment of the requirements is analyzed; identifying the sources of requirements—where the origin and reasons of the requirements are determined; analyzing the stakeholders—where all stakeholders relevant to the requirement are identified; selecting the techniques, approaches, and tools to use—where the best possible strategy for requirements gathering, e.g., interviews or workshops, is determined; and eliciting the requirements from stakeholders and other sources—where the actual requirements-elicitation activity is conducted.

The role of the requirements engineer has a central meaning within the process since this person acts as a facilitator, mediator, and manager of the actual requirements, including analysis and documentation [6].

Requirements can generally be classified differently. The main types of requirements are functional requirements—requirements and tasks to be accomplished by a system,

and quality requirements—also referred to as ‘constraints’ in literature—that describe the system’s behavior [7].

2.2. Artificial-Intelligence-Supported Requirements Elicitation

Recently, many researchers have conducted studies applying artificial intelligence methodologies in requirements elicitation [8–16]. Various methods for automated requirements elicitation are applied in the literature and often include terms such as deep learning [17,18], sentiment analysis [19,20], or automatic text classification [21–23]. Artificial intelligence offers the advantage of efficiently analyzing large amounts of data. However, automatically extracting requirements from text data, such as online reviews, still requires human input. Moreover, [24,25] suggest human reviews of the results of automated requirements elicitation to meet quality expectations and formulate adequate requirements.

In 2011, Lee and Bradlow [16] proposed a method facilitating text mining to support the identification and visualization of product attributes and attribute dimensions among market segments from online camera reviews. By comparing online consumer reviews with expert reviews, the authors showed that consumers and experts value specific product attributes differently. In addition, they determined that segments of users use distinct vocabulary to describe the same features.

Jiang et al. [12] showed an approach to derive requirements from online reviews using automated text analysis and user-satisfaction analysis. They highlighted the challenge of deriving constructive feedback from user reviews due to complex and diverse opinions. To overcome this challenge, the researchers first adopted opinion-mining techniques such as the ‘syntactic relation-based propagation approach’ [26] to automatically extract opinion expressions and then perform clustering of the terms using macro network topology. Finally, they combined user satisfaction analysis with inherent economic attributes.

Rajender et al. [8] introduced an artificial-intelligence-based chatbot to interact with users to elicit requirements. By integrating a knowledge base that provides domain-specific requirements from various industries, the chatbot can ask questions to formulate adequate requirements. Finally, the captured requirements were classified into functional and non-functional requirements using machine-learning algorithms such as support vector machine and multinomial naïve Bayes. According to the research results, the classification achieved 88% and 91% accuracy, respectively.

Other researchers, such as [10,11,14,24,25], propose using the latent Dirichlet allocation (LDA) algorithm [27,28] to extract topics from online reviews automatically. In the studies, the researchers conducted sentiment analysis to determine user satisfaction to classify the requirements into bugs and features. In addition to using LDA and sentiment analysis, Zhang et al. [15] utilize an adapted version of the Kano model [29] to categorize the customer requirements based on the determined dissatisfaction–satisfaction pair from the sentiment analysis. Gülle et al. [11] combined LDA with word-embedding [30] algorithms, such as word2vec [31] and word mover’s distance [32], to summarize requirements automatically.

While Van Vliet et al. [33] achieved state-of-the-art results with traditional methods of classifying new requirements based on online reviews, Mekala et al. [13] showed that the use of modern transformer algorithms such as BERT [34] can reach up to 87% accuracy when taking over manual classification tasks while drastically reducing the time and costs of evaluation.

Zhao et al. [35] proposed, with ReqGen, an approach to automatically generate natural language requirements specifications based on specific given keywords using transformer technology. The method consists of multiple steps, such as domain fine-tuning of the language model, ensuring the occurrence of essential keywords in the generated requirement statements, and requirement-syntax-constrained decoding to close semantic and syntax distances.

Generally, the application of artificial intelligence is not limited to the requirements-elicitation activity within requirements engineering, which underlines its importance to the whole discipline. Various studies in the literature apply state-of-the-art artificial

intelligence methods in other requirements-engineering activities such as requirements specification [36,37], planning [38,39], or validation [40,41].

Artificial intelligence can improve the discipline's performance in combination with existing requirements-engineering approaches such as Patterns of User Experience [42], which describes an analytical framework to create sticky note diagrams from requirements-elicitation workshops. In addition, it can help overcome challenges in global software development by providing analytical insight from mass data and in combination with traditional approaches such as requirements change management following identified success factors from the literature [43].

Besides applying artificial intelligence methods, approaches using traditional software development techniques can also support requirements-engineering activities, which shows that the potential for process improvement is not limited to artificial intelligence only. Among other approaches, Naumchev et al. [44] introduced a new tool, AutoReq, which converts natural language requirements into a format ensuring high-quality requirements, a crucial determinant of software quality. In addition, AutoReq is not limited to the expression of requirements but also offers support in verifying requirements.

3. Proposed Methodology

In this section, the proposed methodology is described. It can be divided into six main steps. This flow is shown in Figure 1, with the colored boxes corresponding to the main steps. For steps two and four, we build upon the theory of Jeong et al. [25] while altering the methods used to more state-of-the-art deep-learning methods. Each step in which a transformer is used is marked with a Huggingface logo because we used the current versions available on the Huggingface website. We also highlight the steps in which human action is necessary.

3.1. Data Retrieval: Extraction of Online Reviews

Online reviews can be retrieved from multiple platforms via APIs or web-crawler technology. For this process, we decided to use data from online marketplaces. For this purpose, a web crawler was developed using Python in version 3.8. We use the Python package Scrapy [45] since it offers an easy implementation. A web link accesses the website concatenated with a unique product identifier to ensure it gets only user reviews related to a particular product. The obtained data contain the review's title, star rating (if available), and the user's comment that is most relevant to the research of that study. Finally, the user reviews are saved in JSON format, which can be easily shared with externals and used in further processing.

3.2. Data Preprocessing and Transformer-Based Key-Phrase Generation

This approach relies on natural language processing (NLP) with transformer models, eliminating the need for text-altering preprocessing steps before applying the models [46]. Nonetheless, the exclusion of not exploitable reviews is imperative. A comprehensive algorithm for eliminating inaccurate comments from the corpus is provided in the form of pseudocode, outlined in Algorithm 1. Sequencing these steps is vital, with language detection, a time-intensive process, strategically positioned at the algorithm's end for each row. This language detection serves a dual purpose: it enhances the performance of transformer models by filtering out non-English reviews and, if a significant proportion of reviews are in other languages, facilitates their translation into English. It is well-known that transformer models consistently yield superior results when applied to English-language content [47]. Any review containing fewer than three words consisting exclusively of alphabetical characters is deemed not exploitable, as such short texts are unlikely to have meaningful information. The identification of these reviews involves employing a regular expression. Additionally, a preprocessing procedure entails eliminating leading and trailing whitespace characters from reviews, as shown in line two of Algorithm 1.

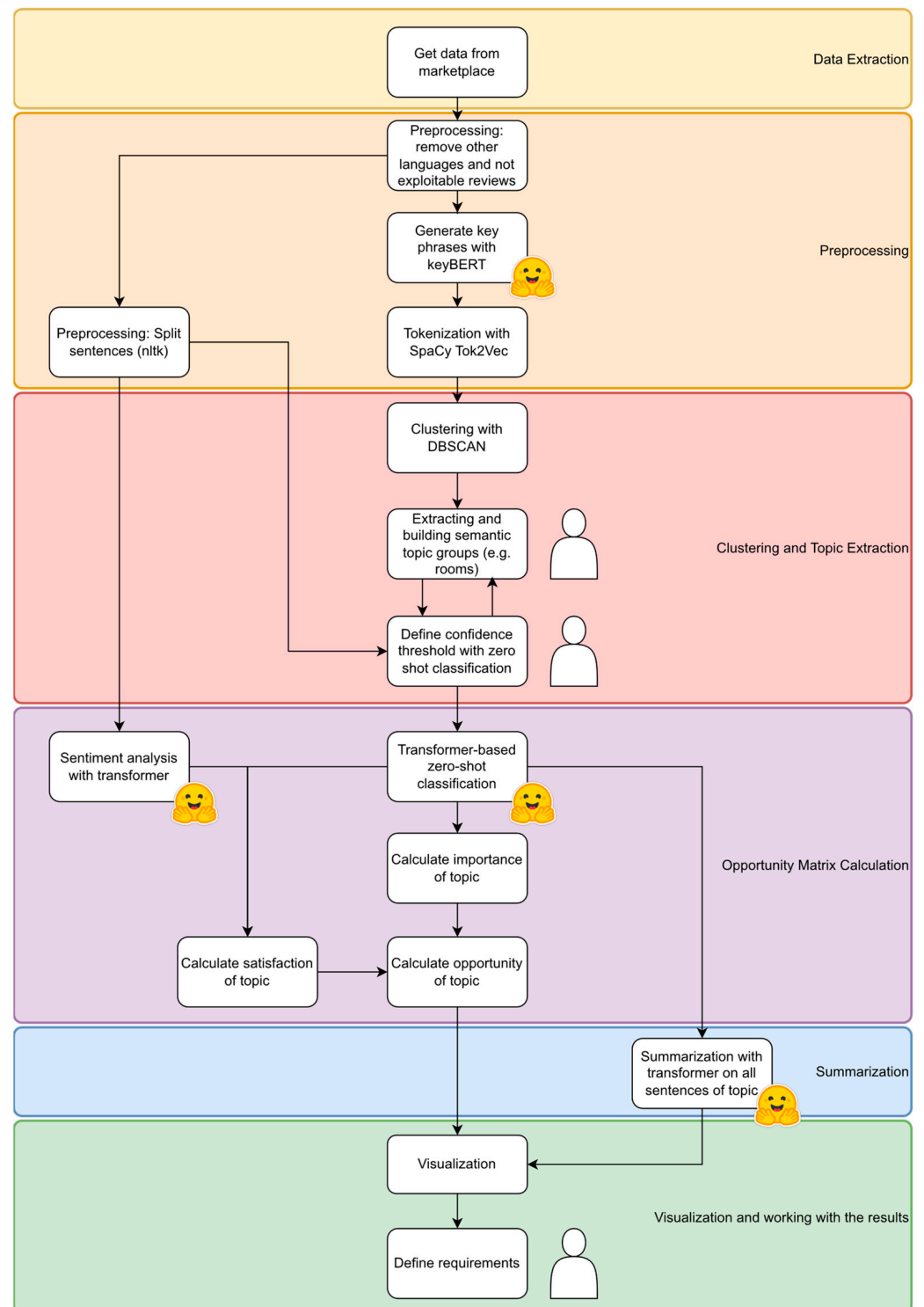


Figure 1. Overview of proposed methodology.

Given that transformer models rely on a predetermined vocabulary from the training data for embedding generation, handling Emojis poses challenges. Emojis are removed using a regular expression to ensure compatibility across multiple models within our proposed methodology. This approach facilitates the application of various models without hindrance.

Algorithm 1: Preprocessing and removal of not exploitable reviews

Input: df_review_texts = DataFrame with all crawled raw reviews
Output: Cleaned DataFrame ready for keyword generation and sentence split

```

1   for comment in df.review_texts:
2       trim comment
3       if comment contains three complete words:
4           regex.remove(emoji_pattern, comment)
5           if spacy.language_detect of comment is 'en':
6               keep comment
7           goto next comment
8       delete comment

```

For keyword generation, a transformer is fed with complete reviews. Since reviews typically employ simple language and short sentences, the transformer with the highest semantic search score should be utilized. A practical approach involves using the keyBERT Python library and a sentence-transformer model surpassing the performance of RAKE, SG Rank, and TextRank [48]. keyBERT extracts keywords and phrases from sentences, with key phrases providing a more comprehensive description of natural language texts than single keywords [48]. The process follows the recommendation of [49] to prepare keywords and key phrases for clustering using a pre-trained Tok2Vec model, as it captures the word similarity required to generate clusters containing keywords belonging to the same topic. To maintain consistent vector sizes for keywords and key phrases, the vectors of all words within key phrases are averaged [49], enabling the calculation of their similarity to keywords. Since no duplicates of crucial phrases and keywords are necessary for clustering them, only distinct key phrases and keywords are used. Also, the performance is increased when less data are used for tokenization and clustering.

The reviews are split into single sentences using the NLTK library to enable the transformer to conduct the sentiment analysis and classification.

3.3. New Approach to Topic Modeling

Transformer-based zero-shot classification can predict to which degree an input text belongs to which label without training a new classification model [50]. Therefore, any clustering algorithm could identify the topics and assign the text to the topics using zero-shot classification. To reduce noise, it is proposed that DBSCAN be used for identifying topics.

After extracting the key phrases, the DBSCAN algorithm is applied to the vectorized key phrases to group them into topics. The DBSCAN algorithm was chosen due to its efficiency with extensive data collections. It identifies and eliminates noise instead of adding it to the nearest topic. It has only two parameters that need to be adjusted to improve the result: a distance epsilon and a minimum number of points that can make up a cluster [51]. A heuristic can be applied to determine a valid combination of the two parameters. The clustering result is acceptable when the noise cluster contains no more than 30% of all data elements and the most prominent cluster has 50% of all data elements [52].

The algorithm used for adjusting the parameters of the DBSCAN is based on the heuristic proposed in [52]. The start parameters chosen for MinPoints are four and twice the dimensions of each datapoint minus one, as suggested in [52]. Starting with four on the lower end adds to the heuristic proposed in [52]. Evaluating the potential MinPoints from both perspectives, i.e., a high and a low starting value, facilitates a more expedient determination of optimal parameters. For each of these MinPoints values, a nearest-neighbor plot is generated. This plot can differ depending on the dataset.

The plot pictures the distance of each point to its k-nearest-neighbor. The y-axis represents the distance from the point to its k-nearest-neighbor. The x-axis shows the

index of data points ordered from those with the highest distance to those with the lowest distance.

The authors suggest the following for finding parameters matching the abovementioned criteria: The nearest-neighbor plots are generated with the start parameters chosen for MinPoints. The y-intercept of the elbows point in the plot is used as the epsilon parameter. If the quality criteria are not met with the resulting clusters, the lower start value for the MinPoints parameter is doubled, the higher start value is halved, and the clustering is done again. The described procedure is repeated until the criteria are met. If the criteria are not met, the algorithm is stopped when the repeatedly doubled low value for MinPoints becomes higher than the repeatedly halved high value for MinPoints. In that case, the MinPoints value with the best values for the quality criteria is chosen. Then the epsilon parameter can be readjusted using this value for the MinPoints parameter.

An automation search of these parameters based on our process is partly possible. The MinPoints parameter could easily be chosen automatically with the algorithm described above. The epsilon parameter is more complex since the resulting plots differ from dataset to dataset. However, this is not part of this study and could be done in future work.

The resulting clusters are then analyzed in a manual reading process and grouped into sensible topics based on a human assessment. The zero-shot classification model returns a confidence score expressing the confidence with which it assigned a topic to a sentence. The confidence for a label decreases with an increasing number of labels used as input for the model when the multi-label option is not used. When using the multi-label option, the result was that some labels were overrepresented. Assessing the labels in the Echo Dot dataset, especially the smart-home label, returned high confidence for most sentences. Therefore, the usage of zero-shot classification without a multi-label option is suggested. A high number of labels for one zero-shot classification pipeline without multi-label is not recommended due to decreasing confidence. However, one sentence could contribute to similar and different topics, but different requirements could be identified.

Examples of that conflict are the topics “bedroom” and “alarm.” The topics are grouped to get the best probabilities. The best results are achieved when these groups contain topics belonging to a semantic group but separate similar topics. The example bedroom should be part of the semantic group “rooms”, and the alarm should be part of the semantic group “clock functions”. Consequently, these topic groups reduce the risk of false multiclassification.

All groups should be about the same size so that the distribution of probabilities does not differ much for each group. Finally, a limit of the probability for assigning a sentence to a topic must be set. This is done by testing different probability levels and filtering a smaller number of sentences. By manually reading the groups, the probability level can be set. For zero-shot classification, even the topic’s name must be chosen wisely, so different names should also be tested in case of poor classification results. A critical challenge in achieving accurate classification is misclassifying sentences unrelated to the designated topic, even when classified with high confidence. To illustrate this, the following paragraph utilizes examples of the case study described in this paper.

For instance, within the “smart home” category, the “light switch” topic highlights features enabling users to control their smart-home lighting via Alexa. However, when using the term “lights” for classification, sentences referring to the light ring atop the Echo Dot, which is relevant to the design, are also classified under it. Another issue arises when pertinent sentences are not detected at all. Within the same “smart home” category, the topic meant to describe temperature-control features exemplifies this. Using “temperature” as a classifier inadequately classifies sentences, while “thermostat” performs effectively. The classifier’s name in zero-shot classification significantly affects the resulting sentences. If a topic should correlate to a specific product, its name should explicitly denote it. For instance, the “TV” topic encompasses all television-related sentences, spanning Amazon’s Fire TV, controlling Apple TV, or voice recognition with active TV audio. Such delineation is necessary when the TV is a principal product feature,

warranting its topic group. Our study categorized “TV” under “media,” with product distinctions clarified in the summary.

In the case study, 6 out of 39 topic names had been reworked so that the classification worked adequately. A maximum of 3 different topic names had to be tested. The zero-shot classification is applied to all sentences once the last topic names, groups, and probability limit are found.

3.4. Opportunity-Matrix Calculation Utilizing Zero-Shot Transformers and Transformer-Based Sentiment Analysis

In the next step, each topic is assigned an opportunity score. The opportunity score consists of importance and satisfaction [25]. The importance is measured by the number of times the customers mention a topic and then normalized on a scale from one to ten [25]. Calculating the customer satisfaction score for each topic requires a sentiment analysis to assign a sentiment score. The sentiment for the entire topic is then calculated from all sentences assigned to it and translated to a customer satisfaction score by normalizing it on a scale from one to ten [25].

Each topic is placed along the satisfaction and importance axes to create the opportunity chart that is visualized as an example in Figure 2.

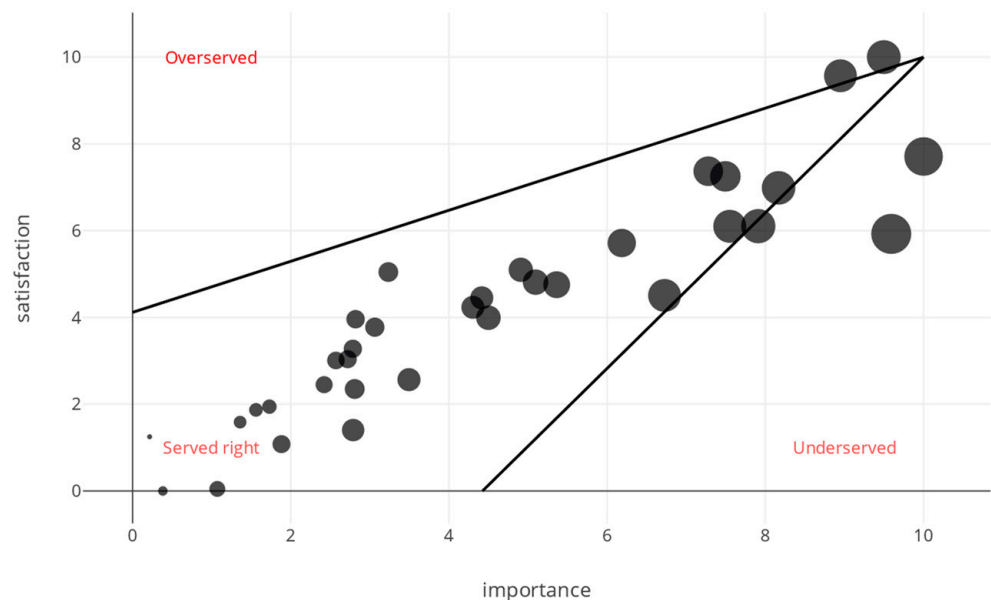


Figure 2. Opportunity chart.

Ideally, each topic has roughly equal importance and satisfaction scores. If a topic has higher importance than customer satisfaction, it should be looked into and improved to increase customer satisfaction [25].

To identify how often a topic is mentioned and to calculate the sentiment of the topic, each sentence from all reviews must be evaluated and mapped to a topic. For this task, a zero-shot classification model is proposed [50,53]. The sentences and the topic names as labels are the input for the matching model, which returns the degree of confidence that a sentence is about the topic. With this confidence score for each topic in each sentence, the importance can be calculated. The importance of each topic is the sum of every confidence score (CS) of this topic in all sentences, normalized from one to ten, considering the sum of

every topic, as shown in Equation (1) [25]. CS_{Min} is the CS of the topic with the lowest CS, and CS_{Max} is the CS of the topic with the highest CS.

$$CS_{Topic} = \sum_{i=0}^{\# \text{ of Sentences}} CS_{Topic, i} \quad (1)$$

$$Importance_{Topic} = 10 \times \frac{CS_{Topic} - CS_{Min}}{CS_{Max} - CS_{Min}}$$

The sentiment of the sentences assigned to the topic can be used to calculate satisfaction. The sentiment analysis is traditionally done on a rules basis, which means that certain words have a more positive or more negative sentiment attached to them and influence the sentiment of the entire sentence [54]. It could cause problems when the sentences contain phrases with negation. Therefore, a transformer model is used in the proposed approach since it overcomes many of the limitations of the Word2Vec and rule-based sentiment classification [55]. The output of the transformer sentiment classification is the probability for three classes—positive, neutral, and negative [56]. For the satisfaction calculation, a sentiment value between one for positive sentiment and minus one for negative sentiment is needed [25]. The probability of negative sentiment is subtracted from the probability of positive sentiment. This sentiment prediction from the transformer classification is introduced as simple sentiment.

The satisfaction score can be calculated with this simple sentiment for each sentence and the mapping to the topics via zero-shot classification. The satisfaction score for topic x in sentence i is the simple sentiment score of this sentence multiplied by the CS of x for sentence i . This way, the simple sentiment score of sentence i is only considered to the degree to which i belongs to topic x . SS for each topic is again the sum of every satisfaction score of this topic in all sentences, normalized from one to ten, considering the sum of every topic, as shown in Equation (2) [25]. SS_{Min} is the SS of the topic with the lowest SS, and SS_{Max} is the SS of the topic with the highest SS.

$$SS_{Topic} = \sum_{i=0}^{\# \text{ of Sentences}} SS_{Topic, i} \quad (2)$$

$$Satisfaction_{Topic} = 10 \times \frac{SS_{Topic} - SS_{Min}}{SS_{Max} - SS_{Min}}$$

Before applying the normalization of CS and SS, the distribution of these scores needs to be checked. If their minimum and maximum value is too close, a normalization from one to ten might not be ideal to represent the difference between those values numerically. It needs to be decided case-wise. However, visualizing the data in the opportunity chart normalization would still be suitable since it shows their relative difference.

Finally, the opportunity can be calculated, and visualization in the opportunity chart, as shown in Figure 2, can be done. Equation (3) [25] shows how the opportunity is calculated.

$$Opportunity = Importance + \text{Max}(Importance - Satisfaction, 0) \quad (3)$$

Algorithm 2 shows all steps for calculating importance, satisfaction, and opportunity. In addition to that, it is also optimized to allow recalculation of the normalized scores when choosing to compare only specific topics. We refer to Section 3.6 for more information and explanation.

Algorithm 2: Calculation of importance, satisfaction and opportunity

Input `df_sentence` = Dataframe with contribution scores of every topic and sentiment score per sentence
 `topic_list` = List of topics that should be considered when calculating importance, satisfaction and opportunity
 `base_importance` = Empty/None
 `base_satisfaction` = Empty/None

Output `df_opportunity` = Dataframe with importance, satisfaction and opportunity score of every considered topic

```

1  if base_importance is None and base_satisfaction is None:
2      importance = empty dictionary
3      satisfaction = empty dictionary
4      for topic in df_sentence:
5          topic_importance = 0
6          topic_satisfaction = 0
7          for sentence in df_sentence[topic, sentiment]:
8              topic_importance += contribution_of_sentence
9              topic_satisfaction += contribution_of_sentence * sentiment_of_sentence
10             importance[topic] = topic_importance
11             satisfaction[topic] = topic_satisfaction
12         base_importance = copy of importance
13         base_satisfaction = copy of satisfaction
14     importance = copy of all topics in base_importance which are mentioned in topic_list
15     satisfaction = copy of all topics in base_satisfaction which are mentioned in topic_list
16     get maximum and minimum value of importance and satisfaction of topics
17     for each topic in topic_list:
18         importance[topic] =  $10 * ((\text{importance\_of\_topic} - \text{minimum\_importance}) / (\text{maximum\_importance} - \text{minimum\_importance}))$ 
19         satisfaction[topic] =  $10 * ((\text{satisfaction\_of\_topic} - \text{minimum\_satisfaction}) / (\text{maximum\_satisfaction} - \text{minimum\_satisfaction}))$ 
20     opportunity = empty dictionary
21     for each topic in topic_list:
22         opportunity[topic] = importance_of_topic +  $\max(\text{importance\_of\_topic} - \text{satisfaction\_of\_topic OR } 0)$ 
23     create dataframe with importance, satisfaction, and opportunity of each topic

```

3.5. Summarization with Transformer

After the zero-shot-classification pipeline has assigned the identified topics as classes to the sentences and the opportunities have been identified, a summarization pipeline is applied topic-wise to sentences where the confidence of the topic assignment is higher than 50%. Humans can then quickly identify what the users have written about the most in this topic. Summarization is a sequence-generation task that copies information from the input but manipulates it [47]. Combining all sentences assigned to a topic in one text corpus and applying a transformer-based summarization pipeline on this corpus, essential passages of the reviews are copied to the output. Since the pre-trained transformer models have a limited input-sequence size, the summaries should be done on the most extended possible sequence of sentences without splitting a sentence. The result is a list of summaries from which a new text corpus can be built that can be summarized again in the most extensive possible input sequence. This summarization can be done recursively until fewer than ten summaries are on the top aggregation level. Based on the author's assessment, ten summaries deliver a good overview of the topics without overwhelming the reader with repeating information.

3.6. Visualization and Working with the Results

The last step of the proposed approach is the visualization and presentation of the results. It is done in an interactive user interface (UI) to combine all the results of the three previous steps. A Streamlit app is suggested because this Python framework runs as scripts

are implemented. Therefore, the previously scripted steps could turn into a UI without expending much development effort. The opportunity chart is the central object of the UI. When hovering over the bubble representing the topics, the details of this topic are displayed. A screenshot of this feature is shown in Figure 3. This figure is created by using the Python library plotly.

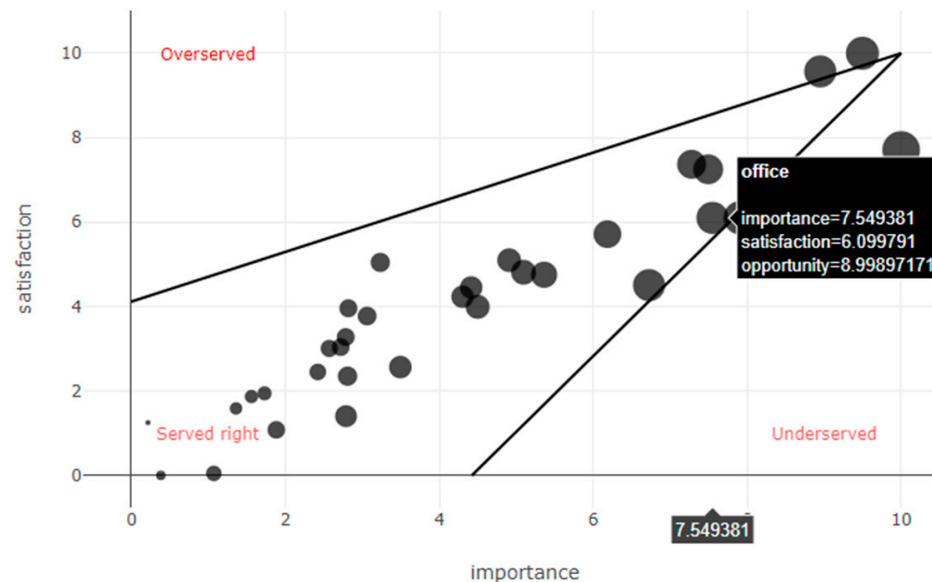


Figure 3. Interactive opportunity chart in Streamlit application.

The related summaries on the highest aggregation level display when the user clicks on a topic bubble. If the information is too highly aggregated, the user can display a lower aggregation level or the original sentences as the lowest aggregation level. This UI element is displayed in Figure 4. The top tabs represent the aggregation levels. The summary consists of different sentences with different sentiments. For a better overview of the sentences, the background color is adjusted to the mean sentiment of all sentences aggregated in this summarization. The color bar on top of Figure 4 shows that bright green indicates the most positive sentiment and solid red indicates the most negative sentiment. If the sentiment reads all positive, but the background color indicates a more neutral or negative sentiment, it is advised to read the lower aggregated summaries.

If the user evaluates one topic as irrelevant or poorly predicted by the zero-shot classifier, meaning the selected sentences are mostly not about the topic, removing the topic from the opportunity chart is possible by removing it from the multi-select box. This box is displayed in Figure 5. After removing a topic, it can be added later using this box.

The importance, satisfaction, and opportunity must be recalculated when the topics have changed since the values are normalized [25] and may change with the selected topics. Algorithm 2 takes this into account and gives an optimized solution for this. It saves the values for each topic before normalizing them so that those base values do not need to be recalculated, but only the normalization needs to be done again. Finally, the boundaries between served-right, overserved, and underserved may change because the mean value of importance and satisfaction determines them. It could lead to new underserved topics with a higher opportunity worth investigating. The proposed UI enables the user to identify and investigate the underserved topics with the highest opportunity using summaries. If the topics are invalid, they should be removed, and the next opportunity should be checked. When the prioritized topics have been selected, the human creative process of requirements engineering [57,58] starts with the acquired information from the presented process. This process should start with underserved topics and topics with the highest opportunity. As input, the user should take the positive and negative sentences from the topic so that human creativity can focus on requirements that reduce unwelcome product

features and improve desirable product features. Since the sentiment analysis is done on a sentence level, the sentiment of the summaries must also be calculated. Applying the sentiment analysis directly to the summaries would result in a loss of information from the underlying original sentences. Therefore, the mean values of the sentiment of all sentences in the summary are used. An advantage of this approach is that the users can directly identify a problem in the summarization. If the reader can interpret all sentences in the summary in a positive sentiment, but the mean value is neutral to negative, the summary does not correctly represent the underlying original sentences.

Selected opportunity: office

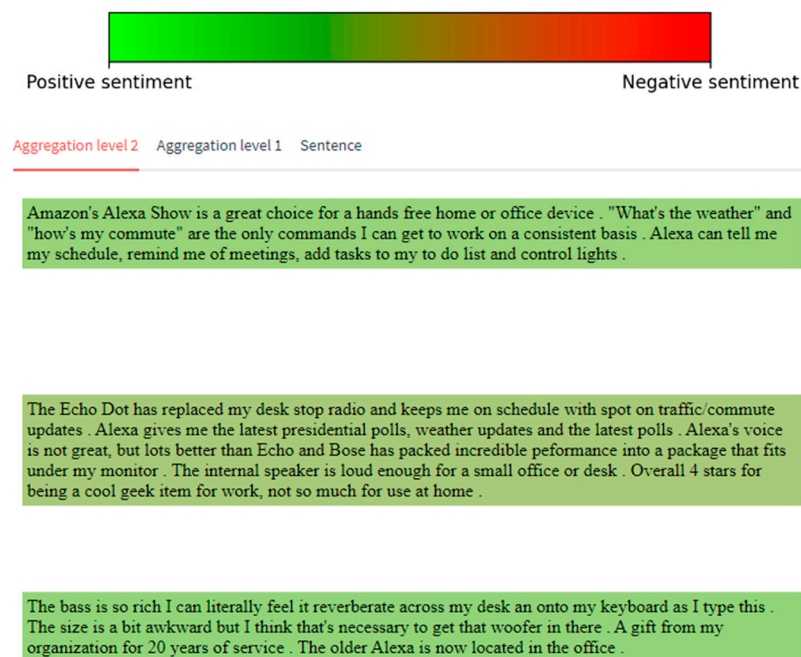


Figure 4. Summary and detailed view on the topic in the Streamlit application.



Figure 5. Multiselect box for topics in Streamlit UI.

4. Case Study: Amazon Echo Dot

This section explains the conducted case study. The case study was executed on the following hardware: Intel(R) Core(TM) i7-6700K CPU @ 4.00GHz 4.01 GHz, 32 GB RAM, NVIDIA GeForce RTX 3080, Windows 10 Home. The dataset used in this paper contains roughly 20,000 Amazon reviews about all generations of the amazon echo dot. The columns in the dataset are the 'title' of the review, the 'comment', and the overall 'rating' in the number of stars. The data element that is the focus of this analysis is the 'comment' or the body of the review since the 'title' often does not contain important information. The

overall ‘rating’ is also not of interest because the goal is to find out what users are passionate enough about to write about, be it positively or negatively.

In preprocessing, only comments containing three words with solely alphabetical characters are kept, and emojis are removed. Afterwards, the language is detected using spaCy and the library `spacy_langdetect`. The comments are filtered to those which were detected as English reviews. These reviews are split into sentences using the `nltk` library, which enables the zero-shot classification pipeline and sentiment analysis.

After preprocessing, the next step toward topics is keyword extraction. The chosen model is `multi-qa-mpnet-base-dot-v1` due to its semantic search performance [59]. The model extracts n-grams of words representing the underlying document, and these keywords do not have to appear in the text next to each other. We set the parameter ‘`keyphrase_ngram_range`’ to (1, 3). The resulting 60,122 keywords are then vectorized into 300-dimensional spacy vectors. The vectorized keywords are clustered into topics using the DBSCAN algorithm from the `sklearn.cluster` package. A heuristic was applied to find a valid combination of the parameters epsilon and MinPoints [52]:

Following the process described in Section 3.3, the start parameters chosen for MinPoints are 4 and 599. For each of these MinPoints parameters, a nearest-neighbor plot, as shown in Figure 6, is generated. In the case study, two elbow points are visible. Both are used for DBSCAN clustering, and the results are checked against the quality criteria. The noise cluster should not exceed the size of 18,037, which is 30% of all data.

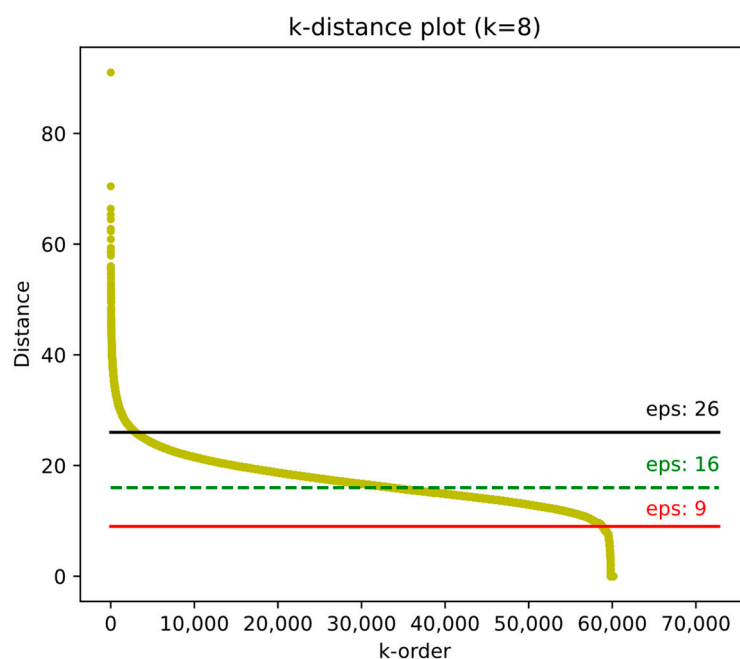


Figure 6. Determination of possible values of DBSCAN-parameter eps for MinPts = 8.

Moreover, the largest cluster must not contain more than 50% of all data and so must not exceed the size of 30,061. Neither of these parameters matched the criteria.

The best result was achieved with MinPoints 8, so the epsilon parameter is adjusted while using this value for the MinPoints parameter.

The highest elbow point of 26 produced one cluster too big to meet the quality criteria, but the amount of noise fulfilled the criteria. The lower elbow point at nine produced too much noise to meet the quality criterion, but the cluster size of the largest cluster met the criterion. Hence the parameter must be set somewhere between these epsilon parameters. The parameter eps was set to 16 and MinPts to 8 to meet the experiment’s quality criteria of DBSCAN clustering.

The resulting 140 clusters were then exported into an Excel file and evaluated manually. The clusters were grouped and filtered in a manual reading process until 39 sensible topics were identified and named.

Before all reviews were fed into the *facebook/bart-large-mnli* model [60,61], tests were done with a small sample group of 1000 reviews to determine a way to get additional results that could be processed and deliver valuable insights. When the model was fed with just the 39 keywords and all sentences from the reviews, the classification confidence was too low to be sure of the dominating topic of the sentence. After some experimentation, the 39 keywords were grouped into 7 keyword classes, as seen in Table 1.

Table 1. Seven keyword classes with corresponding keywords.

Topic Group	Topics
Clock	Alarm, clock, reminder, timer, calendar
Media	Books, podcasts, radio, TV, music
Room	Bedroom, kitchen, office, living room, bathroom
Connection	Connection, Bluetooth, integration, Wi-Fi, set up, app
Design_mobile_sound_voice	Design, portable, battery, bass, sound, voice recognition
Skills_competitors	Shopping, news, weather, google, apple
Smart home	Light switch, smart plug, thermostat, doorbell, hands-free, home automation, security camera

To receive a classification consistent with human judgment, *facebook/bart-large-mnli* worked best if it received around five to seven possible labels to assign to the different sentences. With the confidence score of the topics, the importance is calculated. With CS_{Min} being 4019 for the topic “Wi-Fi” and CS_{Max} being 89,665 for the topic “set up,” a normalization between those values is suited to comprehensively represent the data. Additionally, there were no extreme outlier lies in the distribution, as Figure 7 shows. Base importance on the x-axis was the CS before normalization.

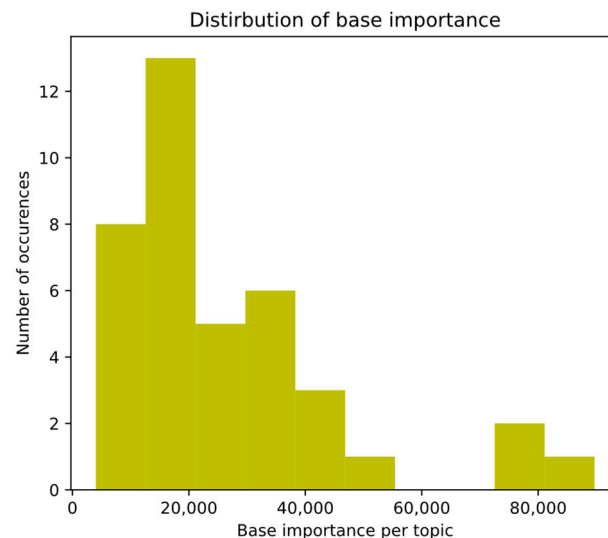


Figure 7. Distribution of base importance.

The model *cardiffnlp/twitter-roberta-base-sentiment* [53] was chosen for the sentiment analysis. Based on the values this model calculates for the review data, the simple sentiment is calculated for each sentence to calculate the satisfaction score. With SS_{Min} being −203 for the cluster “battery” and SS_{Max} being 26,615 for the cluster “sound,” normalization of SS is again suited to represent the data. Like Figure 7, Figure 8 shows no extreme outliers in the distribution. Due to bin width, small negative values are represented in the first bin.

After importance and satisfaction scores were calculated, the opportunity was computed. All values are displayed in Table 2.

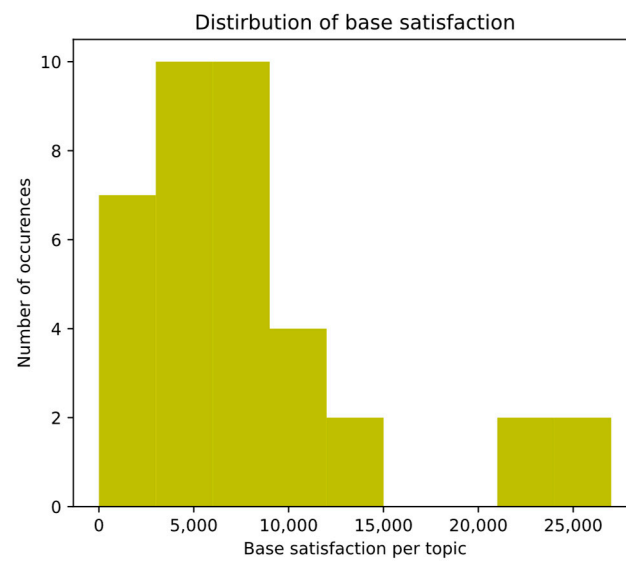


Figure 8. Distribution of base satisfaction.

Table 2. Resulting topics with importance, satisfaction, and opportunity scores and their corresponding segment in the opportunity matrix.

Topic	Importance	Satisfaction	Opportunity	Served
Alarm	2.15	0.74	3.56	Served right
Clock	2.36	2.00	2.75	Served right
Timer	3.79	2.70	4.87	Served right
Calendar	2.18	2.10	2.25	Served right
Connection	3.47	2.12	4.83	Served right
Bluetooth	0.17	0.66	0.17	Served right
Integration	2.49	2.67	2.49	Served right
Wi-Fi	0	0.46	0	Served right
App	1.20	0.99	1.42	Served right
Design	1.33	1.03	1.64	Served right
Portable	7.40	3.14	11.66	Served right
Battery	0.30	0	0.59	Served right
Bass	0.83	0.03	1.63	Served right
Sound	10	10	10	Served right
Voice recognition	5.19	2.39	7.99	Underserved
Books	1.87	1.30	2.44	Served right
Podcasts	3.41	2.36	4.45	Served right
Radio	7.72	4.09	11.34	Underserved
TV	6.10	3.23	8.97	Underserved
Music	6.91	5.07	8.74	Underserved
Bedroom	6.30	3.70	8.90	Underserved
Kitchen	4.77	3.03	6.52	Served right
Office	5.83	3.23	8.42	Underserved
Living room	5.78	3.84	7.72	Served right
Bathroom	3.32	2.24	4.40	Served right
Shopping	2.15	1.74	2.56	Served right
Weather	2.10	1.61	2.59	Served right
Google	3.93	2.55	5.31	Served right
Apple	1.98	1.59	2.37	Served right
Light switch	2.70	1.36	4.03	Served right
Thermostat	2.17	1.25	3.09	Served right
Doorbell	1.33	1.03	1.64	Served right
Hands-free	4.13	2.52	5.76	Served right
Home automation	5.61	3.91	7.33	Served right
Security camera	1.05	0.84	1.26	Served right

After the zero-shot classifier assigned each topic's confidence score to the sentences, the summarization pipeline was applied topic-wise to sentences with a corresponding confidence score higher than 0.5. This score was determined by reviewing the results from classifying the first 1000 reviews.

An example text with approximately 300 words was taken from one of the topics to select the best-performing model. Three hundred words were the limit since it could be read manually, and therefore the authors could assess the summarization, and 300 words fit in tokenizers with a limit of 512 tokens. Since the prototype analyzed product reviews, pre-trained models with reviews were used, and BART was chosen since it outperformed other models in summarization tasks [62]. During these tests, the minimum length and maximum length of the output are also tested since these could differ from model to model. Some models start to repeat the same sentences after a certain length. The best of the assessed models was *sshleifer/distilbart-cnn-12-6*. This model supports an input sequence of 1024 tokens. The default parameter for minimum and maximum length delivered good summarization results. With all parameters set for the summarization, a list of the most prolonged possible input sequences is built to summarize each topic. All these sequences are used as input for the summarization pipeline. If a topic had more than ten summaries, the summaries were summarized to reach a higher aggregation of summaries. Again, the most extended possible input sequence was built from summaries, resulting in an input sequence with a maximum of 1024 tokens. The highest aggregation level was reached after summarizing three times.

As the last step, a simple UI was developed in Streamlit. The user can select the topics which should be displayed in the opportunity chart in a multi-select box. Importance, satisfaction, and opportunity are recalculated, and the opportunity chart is refreshed. The summaries are displayed under the opportunity chart when the user clicks on a bubble in the opportunity chart. As seen in Figure 4, the tabs allow the user to change the aggregation level.

To evaluate the product, the user starts with the most significant opportunities, which are underserved. If many summaries or sentences do not belong to the topic, the topic should be removed from the opportunity chart. A new opportunity chart is then displayed. While reading the summaries, new user needs or bugs can be found and listed for the creative process of requirements engineering.

5. Results and Discussion

This section presents the results of the case study and the identified user needs from the summaries. Through these results, the proposed methodology is tested. In the second part of this section, the proposed methodology is discussed.

5.1. Results for Alexa Echo Dot

With the help of the proposed algorithm and the interactive UI, all topics were examined to assess their quality and identify possible user needs.

The topic "set up" was the only underserved topic when all topics were displayed, but it had to be removed because too many sentences were assigned to it that had nothing in common with the setup process. The "news," "reminder," and "smart plug" topics were excluded from further analysis for similar reasons.

These removals had the effect that the opportunity chart, as shown in Figure 9, was visualized. Table 2 displays all topics, which are displayed in Figure 9 with the assigned opportunity, sentiment, and importance scores.

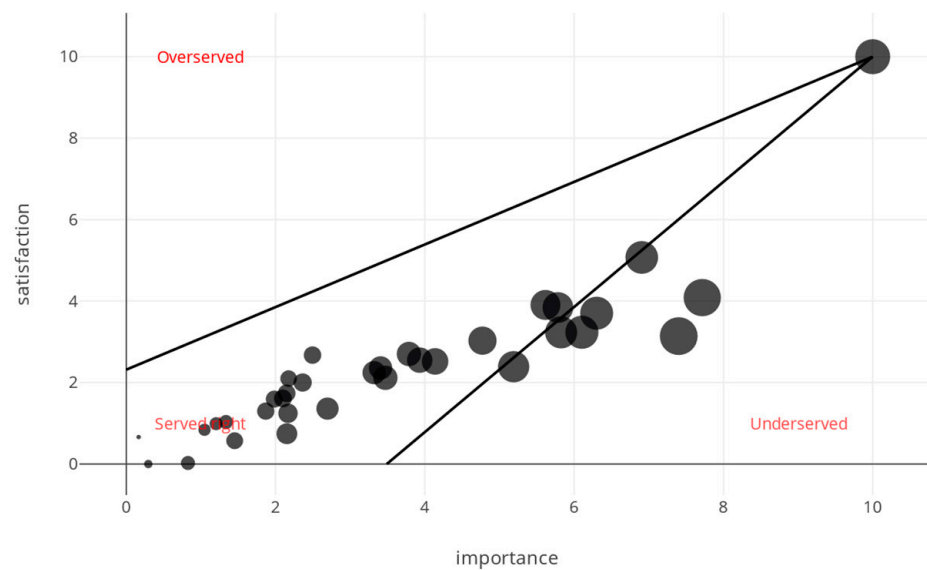


Figure 9. Opportunity chart for Amazon Echo Dot.

The topics that were manually read and analyzed are described below. To improve the comprehensibility of the results, they were divided into three groups: technical requirements, design requirements, and functional requirements. Every topic may have requirements from each category.

5.1.1. Technical Requirements

With Wi-Fi, users raised the issue that the connection to 5 GHz was unreliable and described issues with MAC address authentication, the weak Wi-Fi antenna, and the general first setup. These are all bugs that should be addressed in future generations.

“Connection” is a broad topic that covers issues with Wi-Fi—the echo dot automatically connects to a random network after a power loss instead of the one chosen by the user, connection issues that can only be resolved by unplugging and replugging the device, and the fact that there is no offline mode and an internet connection is essential for Alexa to work. When the echo dot is connected to other speakers via AUX, there will be a slight humming sound when they share the echo power supply. As this impacts the sound quality, this should be fixed in future generations.

For the “sound” topic, the sound quality was a defect many users spoke about—the speaker does not sound good in big rooms, and when the volume is set to more than level eight of ten, the bass output was described as excessive, and users requested a bass filter.

5.1.2. Design Requirements

The sentences on the topic “portable” generally led to the conclusion that portability is a request the customers have that is currently not fulfilled. This topic is heavily interlocked with the requests for a more aesthetically pleasing external battery with a longer battery life than currently available.

“Design” is a topic that is connected to different requirements from different rooms. Users in the office want the Echo to be small enough to fit under their monitor. For more public rooms, the requests were for a better wall mount, a way to hide the cables, choose the cable color, and, maybe, even offer different design packages. The users also did not like that the light-up ring was at the bottom of the Echo in one generation, which makes it difficult to see when it lights up. Another concern with the design was the volume buttons that some users wanted to be replaced by a volume knob.

5.1.3. Functional Requirements

The media topics, such as “radio” and “TV”, contained requests for integrating different streaming services and a more straightforward setup process. Central feature customers were struggling with the fact that while consuming different forms of media voice recognition was impaired by music or the TV playing in the background. The major complaint regarding audiobooks was that the auto-generated voice that reads the Kindle library had only one speed and cadence, which is unpleasant. For podcasts and music, a pain point was that the sound quality was insufficient for a standalone speaker to enjoy the listening experience. Another popular request was better integrating podcast and music libraries, be it a local library from the user’s PC or having a central listening skill combining all media from iTunes, Spotify, and indie podcast publishers.

The topic of “integration” brought more requests for a shuffle function with integrated libraries and a tidal integration, another music streaming platform.

The rooms in the house bring a wide range of different requirements, with the use cases in the bathroom and kitchen needing voice recognition to work with water running in the background. In contrast, as a more extensive space, the living room requires the Echo to produce a higher volume to be heard well. The “living room” topic was also loaded with the “TV” topic and design requirements—users want different color and light options so that their smart speaker can either stand out or blend into their decoration. The “bedroom” topic contained some bug reports about the Echo randomly lighting up, making noise in the middle of the night, and requesting a dimmer function since the standard light was too bright. Users also have Alexa read bedtime stories to their children, which brings the necessity of a child mode with it so that parents can control the content and functions their children can request from Alexa. A functional request from a user outside of the USA regarding the “alarms, traffic and time” function they were using in the kitchen was that Alexa only accepts US postal codes, and therefore the functionality was rendered useless. A complaint made by a user who asked Alexa for the time in the middle of the night when their teenage children left the Echo at top volume led to the conclusion that the volume could either be set to a “night mode” starting at a particular time or that the volume could correspond to the level at which Alexa is spoken.

Issues customers reported with the smart-home features were a bug with the Isteon integration, Alexa having trouble distinguishing between different routines the users set up and mixing up the light bulbs in her control. With the light switch function, users complained about the missing Sengled integration and that Alexa struggles to address light bulbs that were turned off manually. Problems reported with temperature control were that the Echo’s internal temperature sensor delivered inaccurate information and the wish for more compatible thermostat brands. All these issues have relatively apparent solutions. The complaints about “having to use weirdly specific commands” to communicate with Alexa and the wish to have that communication feel more natural require a deeper analysis of the user’s request. It plays into another issue with voice recognition: Alexa cannot correctly interpret and answer all questions directed at her. Users also struggled with editing skills they enabled—to change the skills settings, they had to turn off the skill in the app, modify the attributes, and enable the skill again.

Apart from editing skills, users were generally not satisfied with the app’s experience—they say it is confusing to navigate, and the Amazon ads inside the app were unnecessary. They would also like to access the equalizer settings outside the app.

Voice recognition has been mentioned in many other topics already, but since it is a central function, it has its topic. Ambient noise control has been mentioned in a lot of other contexts. Alexa struggles with children’s voices and heavy accents and recognizing artists’ names when looking for music. A highly requested feature is conversation context. Alexa cannot remember the last things said to her and, therefore, cannot converse. A bug reported in this context was that Alexa wakes up without being spoken to, for example, after a power outage.

A popular request was that Alexa always speaks through the built-in speaker when playing media over Bluetooth—so that the voice input and output do not interfere with the media being played and the users know where always to address Alexa.

The classic assistant functions were generally fulfilled well, but for the clock display, there was a request for the brightness to be adjustable for different rooms, so it could be visible in the kitchen and dimmed in the bedroom. The calendar could be upgraded with a connection to different user profiles and their different calendars. For the alarm function, one user reported a bug that does not let them turn off the alarm via voice command and instead forces them to switch it off in the app. There also seem to be some issues with the snooze function that are not described further. Functions users miss in the alarm feature are cross-device alarms and the possibility to set future, one-time alarms instead of calendar entries.

When compared to competing voice assistants, apple's Siri has better Skype and WhatsApp integration. She can read out and send messages. The "Google" topic has two main subjects. On the one hand, the users request better integration with the Google services such as YouTube and Google Search. On the other hand, Google Home is their home assistant with better voice recognition, better sound quality, and a selection of different voices for the output. Also, the two-word wake-up "Hey Google" is better than the one-word wake-up "Alexa" because the users do not wake up their Google Home by accident as often as Alexa.

Topics customers were generally satisfied with and where no further needs could be derived are the doorbell and security camera integration, shopping and weather services, and the timer function.

5.2. Discussion

In this section, we delve into several important aspects related to our research findings. This section examines the comparison of our study with previous research, especially the work done in [25]. We also explore our significant improvements in requirements elicitation, showcasing the algorithm with its innovative techniques and methodologies. Additionally, we address the limitations inherent in our method, providing a critical assessment of its constraints and potential areas for future improvement. Through these discussions, we aim to comprehensively evaluate our research findings and contribute to the broader understanding of the topic.

5.2.1. Comparison with Previous Studies

Compared to the method described in [25], we incorporated transformers into our process and included transformer-based summarization for each topic. It allowed us to provide detailed information about each opportunity. We achieved better results in extracting opportunities from the review data using transformers. When employing the RAKE algorithm to extract key phrases from our dataset, we obtained 102,850 n-grams with a maximum length of two. However, when using transformers for crucial phrase generation, we obtained 60,122 critical phrases with a maximum length of three. These transformer-generated key phrases required less cleansing than the RAKE key phrases, as they exhibited higher quality. If we had used RAKE with a maximum length of three, it would have resulted in excessive key phrases, requiring more cleansing effort.

Applying the LDA algorithm to the extracted RAKE keywords, as proposed in [25], using bag-of-words encoding resulted in similar topics. Words like "sound", "music", "play", "voice", and "room" were the most common words across almost all topics. Consequently, the topics were too similar to be effectively used in the subsequent steps. Since the generated keywords by KeyBert are not necessarily an exact match with the review text, using LDA with bag-of-words encoding on these keywords is not feasible. To address this issue and the problem of highly similar topics identified earlier, we adopted a topic-generation approach using DBSCAN clustering. However, as the DBSCAN model cannot determine the confidence score of a topic for a document, we employed zero-shot classification with

transformers on the sentences to calculate the confidence score. This calculation is crucial for assessing importance and satisfaction and identifying opportunities.

The Python library *BERTopic* offers topic modeling based on a hierarchical DBSCAN (HDBSCAN) of embedded documents. It also has a variant that uses KeyBert combined with c-TF-IDF, and another variant utilizes zero-shot classification. However, it does not combine both methods. In [63], one of the weaknesses of *BERTopic* is that it allocates each document to only one topic.

5.2.2. Key Improvements to Requirements Elicitation

With our method, this issue is resolved. It is possible to assign one document to several topics, giving a more realistic representation of the mentioned topics. It is also helpful to understand which topics are mentioned together, though this functionality is not implemented in the UI yet.

The extracted user needs show that the proposed algorithm enables users without an information technology background to generate insights from big data with low time investment. The developed UI is a valuable tool for maintaining the connection between topics, summaries, and reviews. It facilitates a better understanding of the results and allows users to perform quality checks by delving into the reviews. The functionality to add and remove topics from the opportunity chart changed the importance, satisfaction, and opportunity of the remaining topics since these values are normalized. Therefore, altering the topics in the opportunity matrix could lead to new, underserved topics. Although there is room for improvement in interface performance and design and the tool has proven indispensable for gathering insights on each identified topic. Without it, retrieving the detailed results of the case study would not have been possible.

During the development of the algorithm, we implemented different language-transformer technologies for the first time in requirements elicitation. We used transformer-generated key phrases for topic modeling. The topic modeling is suggested with DBSCAN to filter the noise in the key phrases. These topics are manually filtered and grouped before zero-shot classification is used to calculate the contribution scores of the topics to the documents. The proposed algorithm adds, with the transformer-based summarization, a completely new idea to requirements elicitation by building summaries from all sentences which have a higher contribution score in zero-shot classification than the defined threshold (0.5 for our case study), and the most relevant information of this topic is extracted in human-readable form. Due to the limitation of input token length to the transformer models, several summarizations need to be done. To present a fast overview, our algorithm suggests summarizing the summaries until there are only ten short summaries so that the human requirements engineer can start with a brief overview of the written texts on this topic.

A lower aggregation level could be used if more information is needed and more summaries are displayed. As the lowest aggregation level, the sentences should be displayed so that the original written sentences, which are not altered by language transformers, can be read. This summarization adds more insights into the identified topics and their opportunities as formerly known algorithms for requirements elicitation. Considering the success factors (SFs) for requirements change management analyzed in [43], the proposed methodology improves the three top SFs. The most essential SF is changing acceptability. It means the product's quality depends on satisfying customer needs and expectations. The proposed solution helps to identify the customer needs and expectations from customer reviews. The second SF is updated requirements. All requirements should be updated since the customers' needs are evolving, and the complete market should also be analyzed. When the proposed methodology is executed periodically, the requirements could be built on the timely needs of the customers. As shown in the case study, it is possible to add topics from competitors to analyze the comparisons with competitors. The third SF, sharing information, could also be improved when the solution is adjusted to this need. Sharing information on existing customer needs is possible if everyone can access the proposed

web surface. However, sharing thoughts and ideas to satisfy them is currently not possible. Adding a comment and networking component to the front end could contribute strongly to this SF.

5.2.3. Limitations of the Proposed Method

While generating the topics from the DBSCAN clusters, the topic “generations” was identified. It represented the generations of the product, and even in the summaries, some comparisons between the generations were found. This topic was dropped since it represents generational differences or development. It would enable filtering the user needs that were fulfilled with a later version. While evaluating the resulting topics and corresponding feature requests, we noticed that some reviews mentioned Google Now—a service that Google Assistant replaced in 2016 [64]. Enabling a software development team to filter out outdated requests, including the posting date in a future export, would be helpful. Having a date and information on the product that was reviewed would help deliver the correct requests to the right product teams. The puck-shaped echo dot has other challenges than the sphere-shaped fifth-generation Echo regarding sound quality, which currently cannot be addressed in the analysis process. Moreover, considering the date could show the importance and satisfaction of the topics over time. Due to new apps, technology, or skills, some user needs might get higher opportunities.

The aggregated summaries delivered requests that were not specific enough to enable a developer to derive an actual solution. Additional engineering activities are necessary, such as requirements specification, planning, and validation. While our method is suitable for requirements elicitation of online reviews, for a more holistic approach to requirements engineering, online reviews should not be the only source of requirements. Furthermore, a transformer model that delivers reasonable requirements summaries could be trained. A model that directly delivers specific requirements from the summaries is desirable. To generate training data, the presented UI could be enhanced. The user could be able to mark the summaries from which a requirement was derived and enter the requirement as text. The training dataset would then consist of two columns: the entered requirement as the label and all combined sentences used for the summary as the text. Another helpful enhancement of the presented UI would be the functionality to drill down into the summaries. It would enable users to see from which summaries the highest aggregated summary was created, and a drill-down to sentences would be possible. It helps in understanding the sentiment of the summaries.

Generally, transformer models have shown significant improvement in the past and are expected to continue doing so in the future. This progress may result in the chosen models performing less effectively than newer models. Nevertheless, our proposed methodology would remain valid, and the results would likely improve using more advanced models.

6. Conclusions

In conclusion, the intersection of requirements engineering and semantic technologies marks an exciting frontier for future investigations. The fusion of these disciplines has unveiled new horizons for extracting meaningful insights from consumer reviews. Our proposed approach, which harnesses the power of natural language processing and language-transformer technologies, represents a substantial leap forward in requirements elicitation from online reviews.

Integrating consumer feedback into product design can boost sales performance and competitive advantage and catalyze broader industry advancements. Our automated methodology efficiently dissects online reviews, yielding a more profound comprehension of consumer preferences. This, in turn, fuels innovation in both processes and products, shaping the trajectory of industries.

While our current algorithm showcases significant enhancements, it is a stepping stone toward future investigations. Avenues for exploration include:

User Interface Refinement: Further refinements to the user interface could enhance user interaction and experience, potentially unlocking more actionable insights from the data.

Advancements in Language-Transformer Technologies: The rapid evolution of language transformer technologies promises continued improvements in natural language understanding, potentially leading to more accurate requirements extraction.

Fine-Tuning Algorithm Parameters: Delving into the fine-tuning of algorithm parameters could unveil optimizations that cater to specific industries or domains, making requirements extraction even more targeted and precise.

Incorporating Multimodal Data: Integrating visual data (such as images or videos) with textual reviews could provide a richer and more comprehensive understanding of consumer sentiments and requirements.

Our research bridges raw consumer data and human-readable requirements, fostering a holistic comprehension of consumer needs. This holistic understanding catalyzes the driving of future innovations, influencing individual product development, and shaping the broader landscape of consumer-centric advancements.

As we move forward, these uncharted territories beckon researchers and practitioners alike to embark on a journey of discovery, innovation, and continuous improvement.

Author Contributions: Conceptualization, P.H., S.S., O.J., N.H. and R.B.; methodology, P.H., S.S. and O.J.; software, P.H., S.S., O.J. and N.H.; validation, P.H., S.S., O.J., N.H. and R.B.; writing—original draft preparation, P.H., S.S. and O.J.; writing—review and editing, N.H. and R.B.; visualization, P.H., S.S. and O.J.; supervision, N.H. and R.B.; project administration, N.H. and R.B.; All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Additional data are available upon request.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Alrumaih, H.; Mirza, A.; Alsalamah, H. Domain Ontology for Requirements Classification in Requirements Engineering Context. *IEEE Access* **2020**, *8*, 89899–89908. [\[CrossRef\]](#)
2. Goldberg, D.M.; Abrahams, A.S. Sourcing Product Innovation Intelligence from Online Reviews. *Decis. Support Syst.* **2022**, *157*, 113751. [\[CrossRef\]](#)
3. Lim, S.; Henriksson, A.; Zdravkovic, J. *Data-Driven Requirements Elicitation: A Systematic Literature Review*; Springer: Singapore, 2021; Volume 2, ISBN 0123456789.
4. Horn, N.; Buchkremer, R. The Application of Artificial Intelligence to Elaborate Requirements Elicitation. In Proceedings of the 17th International Technology, Education and Development Conference, Valencia, Spain, 6–8 March 2023; pp. 2102–2109.
5. Pohl, K. *Requirements Engineering: Fundamentals, Principles, and Techniques*; Springer Publishing Company, Incorporated: Cham, Switzerland, 2010.
6. Zowghi, D.; Coulin, C. Requirements Elicitation: A Survey of Techniques, Approaches, and Tools. In *Engineering and Managing Software Requirements*; Springer: Berlin/Heidelberg, Germany, 2005; pp. 19–46. [\[CrossRef\]](#)
7. Henriksson, A.; Zdravkovic, J. Holistic Data-Driven Requirements Elicitation in the Big Data Era. *Softw. Syst. Model.* **2022**, *21*, 1389–1410. [\[CrossRef\]](#)
8. Surana, C.S.R.K.; Shriya; Gupta, D.B.; Shankar, S.P. Intelligent Chatbot for Requirements Elicitation and Classification. In Proceedings of the 2019 4th International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT), Bangalore, India, 17–18 May 2019; pp. 866–870.
9. Khan, J.A.; Xie, Y.; Liu, L.; Wen, L. Analysis of Requirements-Related Arguments in User Forums. In Proceedings of the 2019 IEEE 27th International Requirements Engineering Conference (RE), Jeju Island, Republic of Korea, 23–27 September 2019; pp. 63–74.
10. Zhou, F.; Ayoub, J.; Xu, Q.; Yang, X.J. A Machine Learning Approach to Customer Needs Analysis for Product Ecosystems. *J. Mech. Des.* **2020**, *142*, 011101. [\[CrossRef\]](#)
11. Gölle, K.J.; Ford, N.; Ebel, P.; Brokhausen, F.; Vogelsang, A. Topic Modeling on User Stories Using Word Mover’s Distance. In Proceedings of the 2020 IEEE Seventh International Workshop on Artificial Intelligence for Requirements Engineering (AIRE), Zurich, Switzerland, 1 September 2020; pp. 52–60.

12. Jiang, W.; Ruan, H.; Zhang, L.; Lew, P.; Jiang, J. For User-Driven Software Evolution: Requirements Elicitation Derived from Mining Online Reviews. In Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining, Tainan, Taiwan, 13–16 May 2014; pp. 584–595.
13. Mekala, R.R.; Irfan, A.; Groen, E.C.; Porter, A.; Lindvall, M. Classifying User Requirements from Online Feedback in Small Dataset Environments Using Deep Learning. In Proceedings of the 2021 IEEE 29th International Requirements Engineering Conference (RE), Notre Dame, IN, USA, 20–24 September 2021; pp. 139–149.
14. Joung, J.; Kim, H.M. Automated Keyword Filtering in Latent Dirichlet Allocation for Identifying Product Attributes from Online Reviews. *J. Mech. Des.* **2021**, *143*, 084501. [\[CrossRef\]](#)
15. Zhang, D.; Shen, Z.; Li, Y. Requirement Analysis and Service Optimization of Multiple Category Fresh Products in Online Retailing Using Importance-Kano Analysis. *J. Retail. Consum. Serv.* **2023**, *72*, 103253. [\[CrossRef\]](#)
16. Lee, T.Y.; Bradlow, E.T. Automated Marketing Research Using Online Customer Reviews. *J. Mark. Res.* **2011**, *48*, 881–894. [\[CrossRef\]](#)
17. Dafaalla, H.; Abaker, M.; Abdelmaboud, A.; Alghobiri, M.; Abdelmotlab, A.; Ahmad, N.; Eldaw, H.; Hasabelrsoul, A. Deep Learning Model for Selecting Suitable Requirements Elicitation Techniques. *Appl. Sci.* **2022**, *12*, 9060. [\[CrossRef\]](#)
18. Sainani, A.; Anish, P.R.; Joshi, V.; Ghaisas, S. Extracting and Classifying Requirements from Software Engineering Contracts. In Proceedings of the 2020 IEEE 28th International Requirements Engineering Conference (RE), Zurich, Switzerland, 31 August–4 September 2020; pp. 147–157.
19. Alturaief, N.; Aljamaan, H.; Baslyman, M. AWARE: Aspect-Based Sentiment Analysis Dataset of Apps Reviews for Requirements Elicitation. In Proceedings of the 2021 36th IEEE/ACM International Conference on Automated Software Engineering Workshops (ASEW), Melbourne, VIC, Australia, 15–19 November 2021; pp. 211–218.
20. Franch, X.; Henriksson, A.; Ralyté, J.; Zdravkovic, J. Data-Driven Agile Requirements Elicitation through the Lenses of Situational Method Engineering. In Proceedings of the 2021 IEEE 29th International Requirements Engineering Conference (RE), Notre Dame, IN, USA, 20–24 September 2021; pp. 402–407.
21. Chen, R.; Wang, Q.; Xu, W. Mining User Requirements to Facilitate Mobile App Quality Upgrades with Big Data. *Electron. Commer. Res. Appl.* **2019**, *38*, 100889. [\[CrossRef\]](#)
22. HaCohen-Kerner, Y.; Dilmon, R.; Hone, M.; Ben-Basan, M.A. Automatic Classification of Complaint Letters According to Service Provider Categories. *Inf. Process. Manag.* **2019**, *56*, 102102. [\[CrossRef\]](#)
23. Köhl, N.; Mühlthaler, M.; Goutier, M. Supporting Customer-Oriented Marketing with Artificial Intelligence: Automatically Quantifying Customer Needs from Social Media. *Electron. Mark.* **2020**, *30*, 351–367. [\[CrossRef\]](#)
24. Bhowmik, T.; Niu, N.; Savolainen, J.; Mahmoud, A. Leveraging Topic Modeling and Part-of-Speech Tagging to Support Combinational Creativity in Requirements Engineering. *Requir. Eng.* **2015**, *20*, 253–280. [\[CrossRef\]](#)
25. Jeong, B.; Yoon, J.; Lee, J.-M. Social Media Mining for Product Planning: A Product Opportunity Mining Approach Based on Topic Modeling and Sentiment Analysis. *Int. J. Inf. Manag.* **2019**, *48*, 280–290. [\[CrossRef\]](#)
26. Qiu, G.; Liu, B.; Bu, J.; Chen, C. Opinion Word Expansion and Target Extraction through Double Propagation. *Comput. Linguist.* **2011**, *37*, 9–27. [\[CrossRef\]](#)
27. Blei, D.; Ng, A.; Jordan, M. Latent Dirichlet Allocation. *J. Mach. Learn. Res.* **2003**, *3*, 993–1022.
28. Horn, N.; Gampfer, F.; Buchkremer, R. Latent Dirichlet Allocation and T-Distributed Stochastic Neighbor Embedding Enhance Scientific Reading Comprehension of Articles Related to Enterprise Architecture. *AI* **2021**, *2*, 179–194. [\[CrossRef\]](#)
29. Kano, N. Attractive Quality and Must-Be Quality. *J. Jpn. Soc. Qual. Control* **1984**, *31*, 147–156.
30. Horn, N.; Erhardt, M.S.; Di Stefano, M.; Bosten, F.; Buchkremer, R. Vergleichende Analyse Der Word-Embedding-Verfahren Word2Vec und GloVe Am Beispiel von Kundenbewertungen Eines Online-Versandhändlers. In *Künstliche Intelligenz in Wirtschaft & Gesellschaft*; Springer Fachmedien Wiesbaden: Wiesbaden, Germany, 2020; pp. 559–581, ISBN 9783658295509.
31. Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.S.; Dean, J. Distributed Representations of Words and Phrases and Their Compositionality. *Adv. Neural Inf. Process. Syst.* **2013**, *26*, 3111–3119.
32. Kusner, M.J.; Sun, Y.; Kolkin, N.I.; Weinberger, K.Q. From Word Embeddings to Document Distances. In Proceedings of the 32nd International Conference on Machine Learning, Lille, France, 6–11 July 2015.
33. Van Vliet, M.; Groen, E.C.; Dalpiaz, F.; Brinkkemper, S. Identifying and Classifying User Requirements in Online Feedback via Crowdsourcing. In Proceedings of the Requirements Engineering: Foundation for Software Quality: 26th International Working Conference, REFSQ 2020, Pisa, Italy, 24–27 March 2020; Proceedings 26. pp. 143–159.
34. Devlin, J.; Chang, M.-W.; Lee, K.; Toutanova, K. Bert: Pre-Training of Deep Bidirectional Transformers for Language Understanding. *arXiv* **2018**, arXiv:1810.04805.
35. Zhao, Z.; Zhang, L.; Lian, X.; Gao, X.; Lv, H.; Shi, L. ReqGen: Keywords-Driven Software Requirements Generation. *Mathematics* **2023**, *11*, 332. [\[CrossRef\]](#)
36. Dalpiaz, F.; Dell’Anna, D.; Aydemir, F.B.; Çevikol, S. Requirements Classification with Interpretable Machine Learning and Dependency Parsing. In Proceedings of the 2019 IEEE 27th International Requirements Engineering Conference (RE), Jeju Island, South Korea, 23–27 September 2019; pp. 142–152.
37. Panichella, S.; Ruiz, M. Requirements-Collector: Automating Requirements Specification from Elicitation Sessions and User Feedback. In Proceedings of the 2020 IEEE 28th International Requirements Engineering Conference (RE), Zurich, Switzerland, 31 August–4 September 2020; pp. 404–407.

38. Abadeer, M.; Sabetzadeh, M. Machine Learning-Based Estimation of Story Points in Agile Development: Industrial Experience and Lessons Learned. In Proceedings of the 2021 IEEE 29th International Requirements Engineering Conference Workshops (REW), Notre Dame, IN, USA, 20–24 September 2021; pp. 106–115.
39. Rankovic, N.; Rankovic, D.; Ivanovic, M.; Lazic, L. A New Approach to Software Effort Estimation Using Different Artificial Neural Network Architectures and Taguchi Orthogonal Arrays. *IEEE Access* **2021**, *9*, 26926–26936. [\[CrossRef\]](#)
40. Al Qaisi, H.; Quba, G.Y.; Althunibat, A.; Abdallah, A.; Alzu'bi, S. An Intelligent Prototype for Requirements Validation Process Using Machine Learning Algorithms. In Proceedings of the 2021 International Conference on Information Technology (ICIT), Amman, Jordan, 14–15 July 2021; pp. 870–875.
41. Osman, M.H.; Zaharin, M.F. Ambiguous Software Requirement Specification Detection: An Automated Approach. In Proceedings of the 5th International Workshop on Requirements Engineering and Testing, Gothenburg, Germany, 2 June 2018; pp. 33–40.
42. Gardner, H.; Blackwell, A.F.; Church, L. The Patterns of User Experience for Sticky-Note Diagrams in Software Requirements Workshops. *J. Comput. Lang.* **2020**, *61*, 100997. [\[CrossRef\]](#)
43. Akbar, M.A.; Sang, J.; Nasrullah; Khan, A.A.; Mahmood, S.; Qadri, S.F.; Hu, H.; Xiang, H. Success Factors Influencing Requirements Change Management Process in Global Software Development. *J. Comput. Lang.* **2019**, *51*, 112–130. [\[CrossRef\]](#)
44. Naumchev, A.; Meyer, B.; Mazzara, M.; Galinier, F.; Bruel, J.-M.; Ebersold, S. AutoReq: Expressing and Verifying Requirements for Control Systems. *J. Comput. Lang.* **2019**, *51*, 131–142. [\[CrossRef\]](#)
45. Myers, D.; McGuffee, J.W. Choosing Scrapy. *J. Comput. Sci. Coll.* **2015**, *31*, 83–89.
46. Reimers, N.; Gurevych, I. Sentence-BERT: Sentence Embeddings Using Siamese BERT-Networks. In Proceedings of the EMNLP-IJCNLP 2019—2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Hong Kong, China, 7 November 2019.
47. Goyal, N.; Du, J.; Ott, M.; Anantharaman, G.; Conneau, A. Larger-Scale Transformers for Multilingual Masked Language Modeling. In Proceedings of the RepL4NLP 2021—6th Workshop on Representation Learning for NLP, Online, 6 August 2021.
48. Sharma, P.; Li, Y. Self-Supervised Contextual Keyword and Keyphrase Retrieval with Self-Labeling. *Preprints* **2019**. [\[CrossRef\]](#)
49. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient Estimation of Word Representations in Vector Space. In Proceedings of the ICLR Workshop, Scottsdale, AZ, USA, 2–4 May 2013; pp. 1–11.
50. Ye, Z.; Geng, Y.; Chen, J.; Chen, J.; Xu, X.; Zheng, S.; Wang, F.; Zhang, J.; Chen, H. Zero-Shot Text Classification via Reinforced Self-Training. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Online, 5–10 July 2020; Association for Computational Linguistics: Stroudsburg, PA, USA, 2020; pp. 3014–3024.
51. Ester, M.; Kriegel, H.-P.; Sander, J.; Xu, X. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining, Portland, OR, USA, 2–4 August 1996.
52. Schubert, E.; Sander, J.; Ester, M.; Kriegel, H.P.; Xu, X. DBSCAN Revisited, Revisited: Why and How You Should (Still) Use DBSCAN. *ACM Trans. Database Syst.* **2017**, *42*, 1–21. [\[CrossRef\]](#)
53. Pelicon, A.; Pranjić, M.; Miljković, D.; Škrlić, B.; Pollak, S. Zero-Shot Learning for Cross-Lingual News Sentiment Classification. *Appl. Sci.* **2020**, *10*, 5993. [\[CrossRef\]](#)
54. Mansar, Y.; Gatti, L.; Ferradans, S.; Guerini, M.; Staiano, J. Fortia-FBK at SemEval-2017 Task 5: Bullish or Bearish? Inferring Sentiment towards Brands from Financial News Headlines. In Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017), Vancouver, BC, Canada, 3–4 August 2017; pp. 817–822. [\[CrossRef\]](#)
55. Xiong, G.; Yan, K. Multi-Task Sentiment Classification Model Based on DistilBert and Multi-Scale CNN. In Proceedings of the 2021 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCCom/CyberSciTech), Calgary, AB, Canada, 25–28 October 2021; pp. 700–707. [\[CrossRef\]](#)
56. Kicken, K.; De Maesschalck, T.; Vanrumste, B.; De Keyser, T.; Shim, H.R. Intelligent Analyses on Storytelling for Impact Measurement. In Proceedings of the Sixth Workshop on Noisy User-generated Text (W-NUT 2020), Online, 19 November 2020.
57. Hoffmann, O.; Crolepy, D.; Crolepy, A.; Nguyen, L.; Swatman, P. Creativity, Requirements and Perspectives. *Australas. J. Inf. Syst.* **2005**, *13*, 69. [\[CrossRef\]](#)
58. Suwa, M.; Gero, J.; Purcell, T. Unexpected Discoveries and S-Invention of Design Requirements: A Key to Creative Designs. *Des. Stud.* **2006**, *21*, 297–320.
59. Reimers, N. Pretrained Models. Available online: https://www.sbert.net/docs/pretrained_models.html (accessed on 8 August 2023).
60. Yin, W.; Hay, J.; Roth, D. Benchmarking Zero-Shot Text Classification: Datasets, Evaluation and Entailment Approach. In Proceedings of the EMNLP-IJCNLP 2019—2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Hong Kong, China, 7 November 2019.
61. Tesfagergish, S.G.; Kapočūtė-Dzikiene, J.; Damaševičius, R. Zero-Shot Emotion Detection for Semi-Supervised Sentiment Analysis Using Sentence Transformers and Ensemble Learning. *Appl. Sci.* **2022**, *12*, 8662. [\[CrossRef\]](#)
62. Lewis, M.; Liu, Y.; Goyal, N.; Ghazvininejad, M.; Mohamed, A.; Levy, O.; Stoyanov, V.; Zettlemoyer, L. BART: Denoising Sequence-to-Sequence Pre-Training for Natural Language Generation, Translation, and Comprehension. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Online, 5–10 July 2020; Association for Computational Linguistics: Stroudsburg, PA, USA, 2020; pp. 7871–7880.

63. Grootendorst, M. BERTopic: Neural Topic Modeling with a Class-Based TF-IDF Procedure. *arXiv* **2022**, arXiv:2203.05794.
64. Manuel Bronstein Bringing You the Next-Generation Google Assistant. Available online: <https://blog.google/products/assistant/next-generation-google-assistant-io/> (accessed on 8 August 2023).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.