Xiaotian Bi

Machine Learning NLP-based recommendation system on production issues

School of Technology and Innovations
Master of Science thesis
Industrial Systems Analytics

Vaasa 2023

UNIVERSITY OF VAASA
School of Technology and Innovations
Author: Xiaotian Bi
Title of the thesis: Machine Learning NLP-based recommendation system on production issues
Degree: Master of Science in Technology
Discipline: Industrial Systems Analytics
Supervisor: Mohammed Elmusrati
Petri Välisuo
Year: 2023 **Pages:** 77

ABSTRACT :

The techniques related to Natural Language Processing (NLP) as information extraction are increasingly popular in media, E-commerce, and online games. However, the application with such techniques is yet to be established for production quality control in the manufacturing industry.

The goal of this research is to build a recommendation system based on production issue descriptions in a textual format. The data was extracted from a manufacturing control system where it has been collected in Finnish on a relatively good scale for years. Five different NLP methods (TF-IDF, Word2Vec, spaCy, Sentence Transformers and SBERT) are used for modelling, converting human digital written texts into numerical feature vectors. The most relevant issue cases could be retrieved by calculating the cosine distance between the query sentence vector and corpus embed matrix which represents the whole dataset. Turku NLP-based Sentence Transformer achieves the best result with Mean Average Precision @10 equal to 0.67, inferring that the initial dataset is large enough using deep learning algorithms competing with machine learning methods. Even though a categorical variable were chosen as a target variable to compute evaluation metrics, this research is not a classification problem with single variable for model training. Additionally, the metric selected for performance evaluation measures for every issue case. Therefore, it is not necessary to balance and split the dataset.

This research work achieves a relatively good result with less data available compared to the size of data used for other businesses. The recommendation system can be optimized by feeding more data and implementing online testing. It also has the possibility to transform into collaborative filtering to find patterns of users instead of simply focusing on items, in the condition of comprehensive user information included.

KEYWORDS: NLP, Recommendation System, Mean Average Precision @K, Sentence Transformers, SBERT.

Acknowledgements

I would like to express my gratitude to my academic advisor, Prof. Mohammed Elmusrati and Petri Välisuo, providing incredible support and useful insights to make my thesis research proceeding smoothly.

I also appreciate that Mr. Christian Sundman offered this interesting topic to me, and other employees from company side gave valuable advice from MES and data science point of view.

Contents

1 Introduction 7

2 Natural Language Processing (NLP) and Recommendation System 10

2.1 Natural Language Processing (NLP) 10

2.2 Recommendation System 12

2.3 NLP-based Recommendation system 14

3 Methodology 16

3.1 Content-based Recommendation System 16

3.2 Python Libaries for NLP 20

3.3 NLP Methods 21

3.3.1 Term Frequency-Inverse Document Frequency 21

3.3.2 Word2Vec 23

3.3.3 SpaCy 25

3.3.4 Sentence Transformers 28

3.3.5 SBERT 29

3.4 Evaluation Metrics 31

3.4.1 Decision Support Metrics 32

3.4.2 Ranking-based Metrics 34

3.4.3 Other Metircs 36

4 Case Study 38

4.1 Data Collection and Introduction 38

4.2 Data Pre-processing 39

4.3 Data Analysis and Visualization 41

4.4 Model Training and Prediction 44

4.4.1 Term Frequency-Inverse Document Frequency 45

4.4.2 Word2Vec 46

4.4.3 SpaCy 47

4.4.4 Sentence Transformers 48

4.4.5 SBERT 50

4.5 Results 51

5 Conclusions, Discussions and Future Works 55

References 58

Appendices 61

Appendix 1. Wordcloud for Each Class in “REASON_CLASS” 61

Appendix 2. Distribution of AP@10 for the Models 67

Appendix 3. Distribution of MAP@K based on “REASON_CLASS”(K=10) 71

Abbreviations

AI Artificial Intelligence
MES Manufacturing Execution System
NLP Natural Language Processing
TD-IDF Term Document-Inverse Document Frequency
BERT Bidirectional Encoder Representations from Transformers
SBERT Siamese BERT
P@K Precision @K
AP@K Average Precision @K
MAP@K Mean Average Precision @K
POS Part-of-Speech
NER Named Entity Recognition
NLTK Natural Language Toolkit
CBOW Continuous Bag-of-Words
RNN Recurrent Neural Networks
MLM Masked Language Modelling
NDCG@K Normalized Discounted Cumulative Gain
ARHR Average Reciprocal Hit Rank

1 Introduction

Retrieving relevant information from big data with machine learning model is a popular application in the business environment. Embedding such function into program works as search engine to suggest items that users might be interested in, or new items could have some relation with. The algorithms for recommendation are based on keywords, user activity or other features characterizing users or items that systems serve for. The most common feature is in a contextual format, such as product description, customer reviews, etc. Information supporting recommendation is normally collected from website with massive of data accessible. Therefore, well-developed recommendation system appears more often in the platforms of E-commerce and media business, like Amazon Prime, Netflix, and Google.

In the field of manufacturing industry, quality control is a trivial but difficult topic which determines the reputation of a company and its product. Even though artificial intelligence (AI) is widely involved in many phases of manufacturing, quality control still lacks AI applications to support with less data available. Another reason might be the fact that very few pre-trained deep learning models can be utilized for a narrow scope of industries with terminology words in non-English.

This research is sponsored by a manufacturing company aiming to make its way of working smarter by using AI technology and its history data. The idea is to build a recommendation system for production control system named as Manufacturing Execution System (MES), providing relevant history cases to new issue case according to production issue descriptions recorded by factory employees in a digital written format.

The data has been collected from MES during the period of 11/2016-06/2022 reported by around 60 employees with over 30,000 comments on issue descriptions in the language of Finnish.

Various of Natural Language Processing (NLP) methods are applied in recommendation system modelling, such as TD-IDF, Word2Vec, spaCy, sentence transformers and Siamese BERT (SBERT). NLP-based models enable the machine to search for documents within a dense corpus that are highly related to new issue topic. Semantic similarity is computed to retrieve the most relevant production issues by every single model.

Chapter 2 introduces the concepts which are engaged in the thesis, such as NLP, recommendation system, and NLP-based recommendation system. The methodology chapter describes the theories applied in this research. Content-based recommendation system is further explained, and the principles of five NLP methods are briefly illustrated. Term Frequent-Inverse Document Frequent (TF-IDF) are the baseline of those models by using the frequency of the words and creating vectors for sentences. Word2Vec is a simple neural network with the advantages of detecting analogy. SpaCy provides pretrained pipelines for multiple languages with state-of-the-art performance. Sentence transformers and SBERT are deep learning methods for vectorization, and they contain mature pre-trained models for Finnish language. (LD, 2021) Evaluation metrics are introduced in the methodology chapter as well to validate the performance of the recommendation system. Precision @K (P@K), Average Precision @K (AP@K) and Mean Average Precision @K (MAP@K) are three metrics to evaluate both order-unaware and order-aware lists. In the chapter 4, the actual case study is demonstrated. After data pre-processing and visualization, the NLP-based models are trained and tested by retrieving top 10 relevant issue cases by randomly choosing an issue description. The obtained results are reviewed with the conclusion that deep learning algorithms such as sentence transformers and SBERT achieve best results compared to other methods.

In the end of chapter 4, some proposals are suggested for future optimization, such as reviewing the result with online A/B testing or change target variable with different interests. The recommendation system can be converted into collaborative filtering when the structure of dataset changes with more engagement of system users. The limitation of this research is revealed, leaving improvement space for future investigation. There

were some challenges to confront with during implementation, but the research is finally completed with a relatively good result utilizing a smaller size of data compared with other recommendation systems built from big data.

2 Natural Language Processing (NLP) and Recommendation System

Rooted from 1950s, Alan Turing introduced a criterion of machine intelligence-Turing test, which can measure the humanity of artificial intelligence with the engagement of a computer program and a human judge in a real-time written conversation environment. (A.M.Turing, 1950) As the progress of technology, computer has been developed into understanding not only text but also voice like human. Nowadays in the era of big data from various applications and Internet, people prefer to obtain precise suggestions, assisting them to make efficient decisions from a tremendous number of products and services. Many of the sources that recommendations come from are in textual form, like user reviews, or product descriptions.

2.1 Natural Language Processing (NLP)

NLP, short for Natural Language Processing, evolved from computational linguistics, utilizes approaches from different disciplines, such as computer science, artificial intelligence, linguistics, and data science, enabling computers to understand human language in both written and verbal forms. (Kavlakoglu, 2020)

The research and development of NLP have gone through the phases of symbolic-based, statistical-based, and neural network-based NLP.

During 1950s to early 1990s, the machine can emulate a few functions of NLP tasks given a collection of rules, such as automatic translating a limited number of sentences into another language, therapist simulation working with restricted vocabularies, and conceptual ontologies structured real-world information into computer understandable data. Even though rule-based methods are less accurate than machine learning, symbolic methods nowadays still take vital roles on NLP pipelines such as tokenization, knowledge extraction from syntactic parses. (Wikipedia, Natural language processing, 2023)

Statistical NLP grew rapidly with the development of machine learning algorithms during the period of 1990s and 2010s, which concentrating on building up statistical models, providing soft, probabilistic decisions based on input features from dataset. Compared with rule-based methods, machine translation works more efficiently by applying statistical model on text corpus. With the availability of big data on the internet, statistical NLP models can solve unsupervised or semi-supervised problems analyzing non-annotated data or a combination of annotated and non-annotated data. (Wikipedia, Natural language processing, 2023)

Neural networks NLP, utilizing deep learning instead of machine learning algorithms, becomes relatively popular since 2010s due to state-of-the-art result achievement in modeling and parsing. Machine translation has been shifted to neural machine translation, implementing neural networks approaches on sequence-to-sequence transformations. (Wikipedia, Natural language processing, 2023) Higher-level NLP tasks such as question answering, text summarization, grammatical error correction can be realized by capturing semantic properties with word embeddings or even directly utilizing pretrained model for transfer learning, which are some of popular techniques of neural networks.

NLP consists of enormous practical applications in the field of healthcare diagnoses, cyber security, and online customer services. One of the famous NLP applications is ChatGPT which is an artificial intelligence chatbot developed by OpenAI and launched in November 2022. It uses supervised and reinforcement learning to build and fine-tune large language models, providing solutions to customers across many domains of knowledge. (Wikipedia, ChatGPT, 2023)

Common NLP tasks are listed like following: (Jung, 2021)

- Speech recognition (or speech-to-text): It is the task of converting voice to text.

There are many challenges for task realization, such as the way people talk, grammatical errors and background noise.

- Part of speech tagging (or grammatical tagging): It is a process for determining the part-of-speech (PoS) tagging of a particular word or piece of text based on its use and context.

- Word sense disambiguation: It is the selection of the meaning of a word with multiple meanings through a process of drawing meaning from text.

- Named entity recognition (NRE): It is the process of finding entities that can be categorized into names, locations, percentages, etc.

- Sentiment analysis: Performing opinion mining or emotion analysis from customer reviews and analyzing it with NLP techniques.

- Natural language generation: Conversational systems like chatbots enable to generate text like human produced.

There are many other NLP tasks, like image captioning, question answering and so on, which will not be analysed in this chapter.

2.2 Recommendation System

A recommendation system is a subclass of information filtering system, usually associated with machine learning, that uses big data to suggest or recommend additional products to consumers. (Nvidia, 2023) It is widely used especially in commercial industries by providing personalized suggestions to enhance customer satisfaction, engagement, and sales revenue. To illustrate the importance of recommendation system in real-world, Amazon, Spotify and Google spent much effort on their own systems optimization. Netflix even organized a competition asking for a more advanced recommendation system with 1 million dollars.

To suggest relevant items to users, recommendation systems can be classified into three categories: (Figure 1)

- Content-based filtering
- Collaborative filtering
- Hybrid

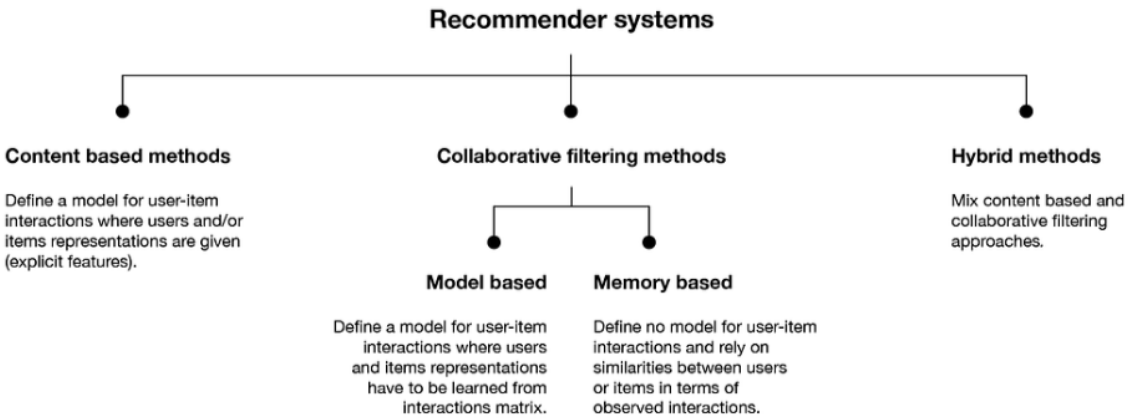


Figure 1. Summary of the different types of recommendation systems algorithms

(Rocca, 2019)

Content-based filtering selects items based on the correlation between the content of the items and this certain user's preferences. For example, if user A read one article, another similar article will be proposed to him or her. This technique can be user or item centered, which will be further illustrated in the methodology session. (Figure 2) Content-based filtering has the advantage of no demand of other users and possibility of niche items recommended, while it requires much domain knowledge and provides outcomes merely based on existing interests of the user. (Krasnoshchok, 2014)

Collaborative filtering chooses items based on the correlation between people with similar preferences. For example, if two articles were read by both A and B, the recommendation system would define A and B as similar users, then the other articles read by A will be suggested to B. Collaborative filtering can be divided into memory-based and model-based approaches depending on whether the model is parametric or non-parametric related. (Figure 2) Compared to content-based filtering, collaborative filtering doesn't need domain knowledge and new interests can be discovered for users. However,

it is purely a cold-start method and difficult to include features beyond query. (Krasnoshchok, 2014)

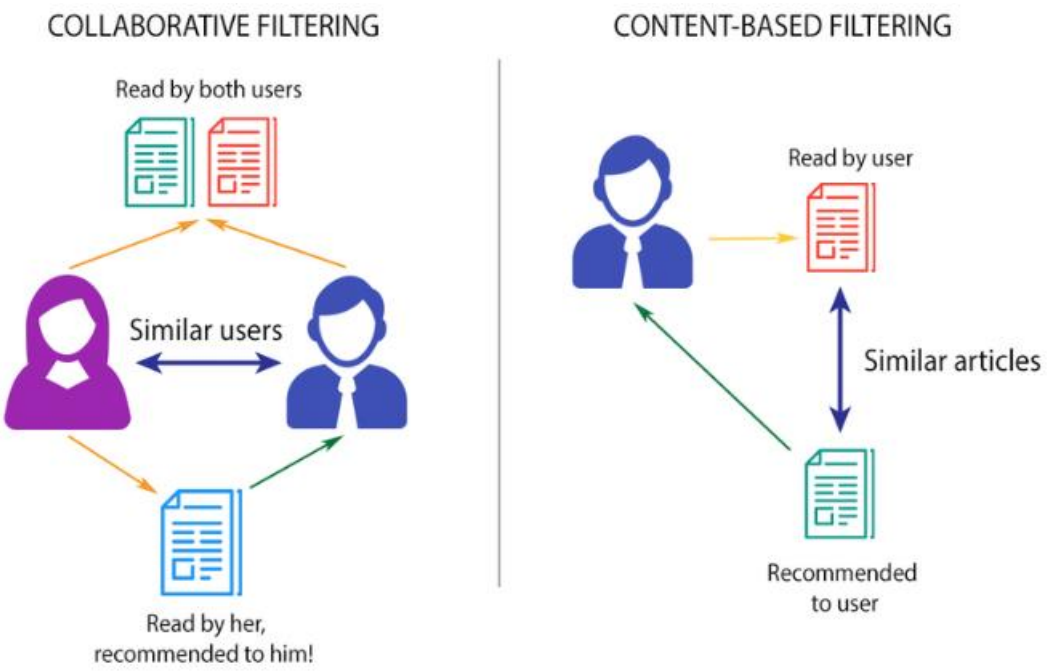


Figure2. Content based methods vs Collaborative filtering methods

(Source: https://www.themarketingtechnologist.co/building-a-recommendation-engine-for-geek-setting-up-the-prerequisites-13/)

Hybrid recommendation system is the combination of content-based and collaborative filtering methods, which possesses the advantages of content-based and collaborative filtering, generates a more comprehensive system, and exhibits higher accuracy.

2.3 NLP-based recommendation system

Since the recommendation system is to suggest relevant items to users based on their interests, the elements which can reflect users' preferences or item's attributes would become crucial to analyse. Numerical data such as rate of a product, age of user or years of working experience are normally easy to be handled, however, in the real-world the most meaningful feature for recommendation systems is textual data, like user reviews, product descriptions and audio messages in a conversation platform. Therefore, NLP

methods serve for data pre-processing, model training and prediction on textual related input.

Pretrained models which have been previously trained on large dataset are popular to implement as neural networks NLP boosted lately. Models from spaCy library based on convolutional neural networks outperform a pipeline of functions for text processing and understanding. Transformers, BERT (Bidirectional Encoder Representations from Transformers) and GPT-3 represent excellent performance on NLP common tasks, reducing computational cost and time, and saving the resources of building up a model from scratch. Applying those deep learning methods enables the recommendation systems working on the datasets in different languages without intentionally translation into English.

3 Methodology

Since the research is to find approaches retrieving relevant items based on textual descriptions, the recommendation system is content based and implemented in Python by the means of various NLP models. It is always good to quantitively evaluate the performance of the models; thus, several evaluation metrics are introduced in this session.

3.1 Content-based Recommendation System

The content-based filtering is according to description of the items and recommends relevant items to user. Unlike collaborative filtering methods, it has no restrictions of cold start problem, which means that it can provide suggestions related to new entities simply according to the features of the new items or users. The content-based methods have the root of information retrieval and information filtering research, and it can be categorized into item-centered and user-centered. If recommendation system uses similarities between new item and previous items for suggestion, it can be defined as item-centered, while recommending items by collecting user's information is user-centered. In this research, the recommendation system is content-based item-centered filtering due to production issue description in the dataset as single feature space to compare similarities with.

Selecting an appropriate similarity metric is the essential part of setting up a successful recommendation system. There are multiple similarity metrics, and four commonly used ones are briefly introduced in the thesis. (Deutschman, 2023)

- Cosine similarity: This metric is to measure the similarity between two vectors of an inner product space, which is widely used in high-dimensional features, especially in text mining.

$$cosine(x, y) = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2} \sqrt{\sum_{i=1}^{n} y_i^2}}$$

From the formular above, we can see that cosine similarity can compute the difference between two given documents in terms of directions but not magnitude.

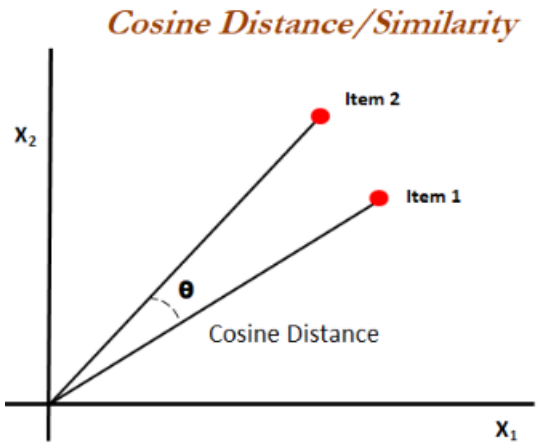
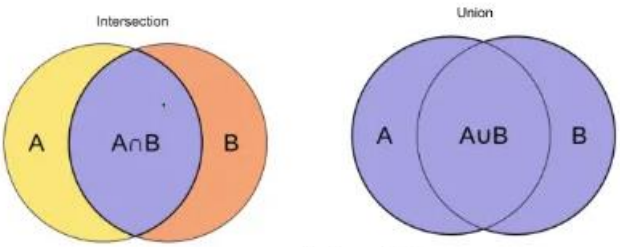


Figure 3. Cosine Similarity

The angle between two item vectors is the visual reflect to cosine similarity, (Figure 3) the smaller the angle, the higher the similarity of two items would be. Converting angle into cosine value, the cosine similarity can be interpreted in another way, that the larger cosine similarity value is, the more likely two items would be alike.

- Jaccard similarity: This is the metric to measure the similarity between two sets of data or binary vectors, dividing the size of the intersection by the size of the union.



$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

Figure 4. Jaccard Similarity (Chandana, 2021)

Jaccard similarity can be used for text mining, semantic segmentation, and recommendation systems. However, it cannot apply to the vectors with rankings or ratings, and it costs more for computation than cosine similarity.

- Euclidean distance: This is the distance metric for the L2-norm of a difference between vectors and vector spaces. It can be applied in a user-centered recommendation system for measuring the similarity of user's preferences.

$$d\left(Person_i, Person_j\right) = \sqrt{(X_1 - X_2)^2 + (Y_1 - Y_2)^2}$$

In the formula above, X and Y represent the scores a user ranks for two items. By calculating Euclidean distance, a preference space is generated between two users. The less distance between User1 and User2 on ratings for two items, the more possibilities that two users have similar tastes. However, Euclidean distance does not consider the correlation between users and draws inaccurate conclusion on similarities. For example, if the distribution of rates given by one user is much different from another one, the Euclidean distance will imply dissimilarity between two users, however users can be likely correlated with consistence on the ranking differences.

- Pearson correlation coefficient: This metric is to measure the slope of the line that represents the relation between two vectors of users' ratings. (Deutschman,

2023)

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2} \times \sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

The coefficient of Pearson correlation ranges from -1 to 1, which 1 stand for fully positive linear correlation, and -1 means fully negative linear correlation. If r equals to 0, there is no linear correlation between two users. With the regression line as the model for prediction, a new rate of a product can be estimated for one user.

Figure 5. Process of recommendation system for cosine similarity comparison

In this research, cosine similarity is the metric selected for sorting and ranking relevant items. Each textual description can be converted into a vector via different NLP methods, and cosine value of the angle between vectors determines the score of two items' similarity. (Figure 5) The recommendation system listed top k (k=10) relevant history cases for new query in a decreasing similarity score order.

Jaccard similarity is another metric for evaluating model performance with Precision @K, Recall @K and F1 score @K. However, it is generally calculated for order-unaware recommendations. If the ranking of relevant items matters, Average Precision @K (AP@K) and Mean Average Precision @K (MAP@K) are computed.

3.2 Python Libraries for NLP

The research for case study was implemented by Python, which can be seen as one of the most efficient and user-friendly programming languages with its large number of open-sources libraries available. Python provides a wide range of NLP tools and libraries to handle various tasks, such as topic modelling, document classification, part-of-speech tagging, word vectors, sentiment analysis, and so on. Here lists some popular ones: (Jung, 2021) (McFarland, 2022)

- Natural Language Toolkit (NLTK): It is an open-source library strong at text processing for classification, tokenization, stemming, tagging, parsing and semantic reasoning.

- SpaCy: It is an open-source library with pre-trained statistical models in over 64 languages, integrating with other deep learning libraries such as PyTorch and TensorFlow.

- Gensim: It is an open-source library originally for topic modelling, and develops into algorithms implementation, such as Word2Vec, FastText, Latent Semantic Analysis, etc. It can be applied to find text similarity by converting words and documents to vectors.

- Sklearn: It is an open-source library for data analysis, solving text classification, regression, clustering, dimensionality reduction, preprocessing and model selection problems.

- Transformers: It is a Python library accessible to pre-trained models dealing with NLP tasks like sequence classification, text generation, and so on.

- Pytorch: It is an open-source library to carry out NLP and computer vision tasks with powerful APIs, ensuring quick processing even working on complex and big data.

In this research, all the libraries above have been employed in the phase of data preprocessing, model training and prediction.

3.3 NLP Methods

To measure cosine similarity of textual data, document representation is a critical step by converting text document into vectors. The baseline model is to regard the document as a bag-of-word without considering the relation of those words within vectors. Therefore, the expression of document meaning and structure, word dependency and sequence becomes insufficient. Computing similarity between two documents will also turn to be difficult in the case of analogy existence. Additionally, other models for document representation are developed, such as Latent Semantic Analysis, Probabilistic Latent Semantic Analysis, Latent Dirichlet Allocation, Random Indexing and Language Models. (Jung, 2021) In this research, vector space model and language models have been applied on document representation.

3.3.1 TF-IDF

Term Frequency-Inverse Document Frequency is the abbreviation of TF-IDF, which is a widely used numerical statistical method in information retrieval by quantifying the importance of each word in the given document relative to a corpus (the total document set). By the means of TF-IDF, textual data is vectorized in the form of numerical value, making the data easier to be analyzed with any programming tools. In a recommendation system, the search engine aims to figure out relevant documents and ranks them in a descending order of similarity score based on a given query sentence. TF-IDF is an efficient method for calculating similarity score in a vector space model. Literally speaking, TF-IDF is a matter of TF and IDF. TF stands for Term Frequency, while IDF represents Inverse Document Frequency.

TF-IDF= TF*IDF

Term Frequency measures how often a term appears in a particular document, which is highly depended on the document length and word generality. (Scott, 2019) However, the length of the document should not become the factor affecting the word importance,

therefore the term frequency is normalized by divided with the total number of terms in the document. Term frequency is calculated for all the possible words in the corpus.

TF (term, document) = (count of term in document) / (number of words in document)

Alike cosine similarity score, the value of term frequency ranges from 0 to 1. Nonetheless, some common appearing words indicate wrong importance merely from term frequency, such as ‘the’, ‘a’, ‘are’. Taking those words into serious account for text analysis will lead to bad results, hence some of them are grouped as “Stop Words” which are normally removed in the data preprocessing phase, while some non-StopWord detected by all the documents are given less weight to reduce their importance in the corpus by calculating inverse documents frequency.

Document frequency measures the importance of documents containing the term in the whole corpus. By inversing the document frequency, most occurring words in all the documents present low discriminative value, ensuring that all the target terms could acquire reasonable weights for further analysis.

IDF (term) = log (total number of documents / number of documents with term)

In some special cases, there might be no document including target term, so the denominator adds 1 to keep IDF formular valid.

IDF (term) = log (total number of documents / (number of documents with term+1))

TF-IDF provides the importance score for each term in the query document by assigning more weight to words that occur only in a document but not too frequently in the rest, (Jung, 2021) under the assumptions that the relevance of a term to the document topic depends on its frequency in the document, while higher frequency of a term appears in

all documents could discriminates poorly between documents. (Robin van Meteren, 2000)

In this research, the recommendation system with TF-IDF model has suggested top k (k=10) items according to the cosine similarity between query vector and documents' vectors from embedded matrix.

3.3.2 Word2Vec

Artificial Neural Networks (ANN) take critical role in feature extraction for NLP tasks. By building dense vectors for each word named as word embeddings, the data becomes more informative by capturing word analogies and word similarities which bag-of-word is incapable to do. Word2Vec embeddings introduced by Tomas Mikolov in 2013, (Mikolov, 2013) demonstrated a simple neural network model can learn high-quality word vectors from relatively huge dataset. The dense vector from Word2Vec embeddings model enables queries and logical reasoning from large corpora. (Hapke, 2019) For example, the vector like "Finland-Helsinki+ Sweden = Stockholm" can be read as "Finland is to Helsinki as Sweden is to Stockholm".

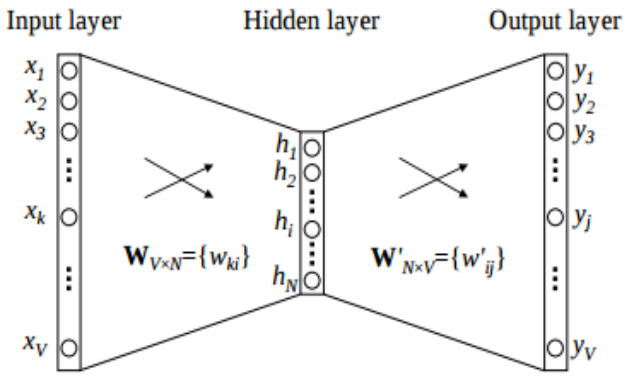


Figure 6. Word2Vec model architecture

Word2Vec model contains an input layer of one-hot-encoded unique words vector, a single hidden layer of word embeddings and an output layer of probability of being neighboring word. (Figure 6) Weight matrix will be extracted from the final hidden layer and perform as word embeddings for test document vector extraction. (Joshi, 2022)

To illustrate Word2Vec better, an example sentence is considered:

The University of Vaasa is an internationally competitive university with a high-level expertise in business, technology, management and communications.

"university" in red is the center word, and the words in blue are neighboring words with a context sliding window size of three. Word2Vec in this case is to calculate the probability of every single word in the vocabulary of being neighboring word in blue.

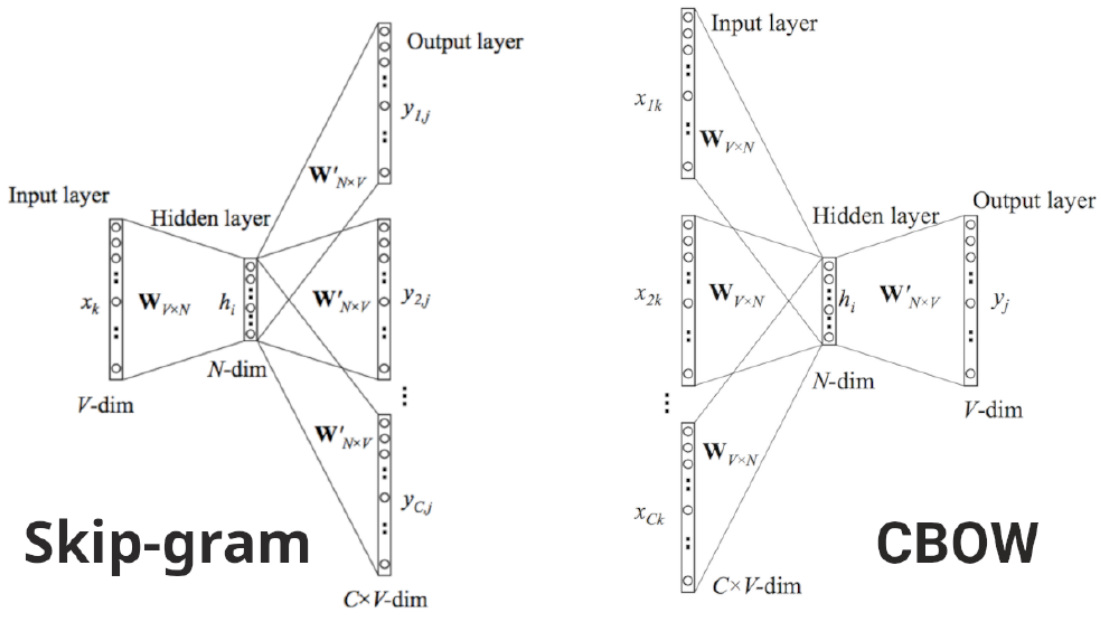


Figure 7. Word2Vec Architectures

There are two models for Word2Vec, Skip-gram and CBOW short for Continuous Bag-of-Words. (Figure 7) Skip-gram is to predict the context by given the centre word, which is suitable for small size of data and uncommon words. Continuous bag of words is to predict the centre word by given its context, which is faster and working well on common words. (Jung, 2021)

In this research, a content-based recommendation system has employed a CBOW model to endorse the top k (k=10) relevant items according to query encoded vector.

3.3.3 SpaCy

SpaCy is a free and open-source library for information extraction and natural language understanding. Although spaCy can independently implement some of its features, loading statistical models enrich the power of spaCy across a variety of languages for performing NLP tasks, such as part-of-speech tagging, named entity recognition, and dependency parsing.

The statistical models are trained pipelines, providing a series of components for linguistic annotations. People might have a better insight on the perspective of contextual data grammatical structure with spaCy models engaged. Spacy models build up a trained pipeline for a text string going through tokenizer, tagger, parser, ner, ending up with an NLP model object. (Figure 8) (Sanagapati, 2020)

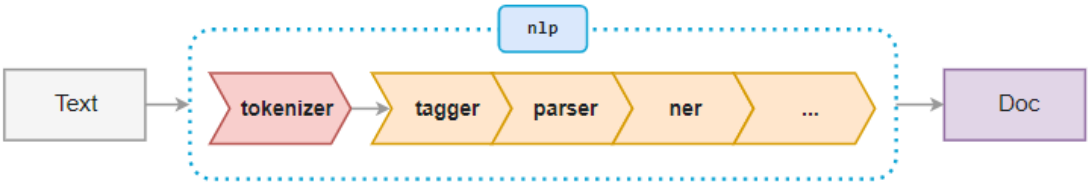


Figure 8. SpaCy's Processing Pipeline

Tokenization is the first step that spaCy model splits text into words based on whitespace characters, specific rules and punctuations. For example, “John can hit the ball.” would be word-tokenized as a list of [‘John’, ‘can’, ‘hit’, ‘the’, ‘ball’, ‘.’]. (Figure 9)

After tokenization, Part-of-Speech Tagging assigns grammatic word tag like noun, pronoun, verb to each token, indicating the word function in the sentence. (Figure 9)

Dependency Parsing identifies the relationship between root word and other words in a sentence. Normally, the first verb in a text can be regarded as root word, since it is close to the beginning of a sentence with less dependence to others and verb is usually more informatively valid to be focused and analyzed. The edges in the figure below represent the grammatical relationships between words and root word. (Figure 9)

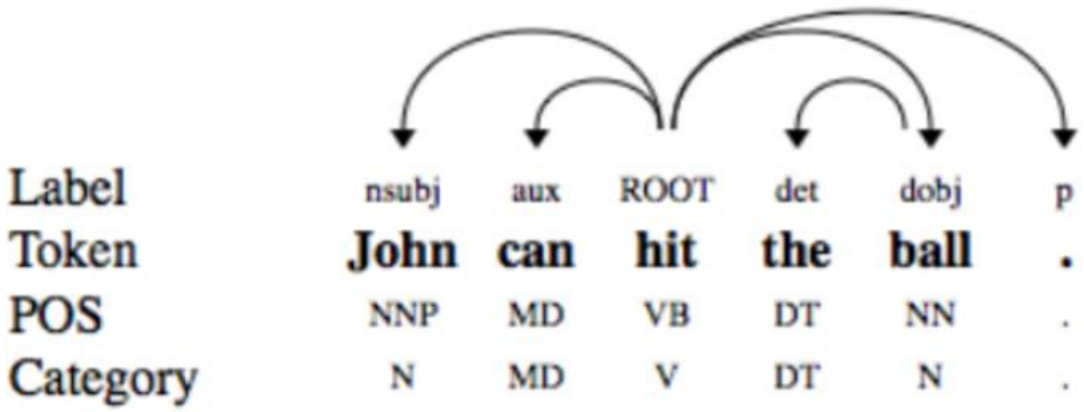


Figure 9. Dependency Parse Tree of an Example English Sentence (Peng Xu, 2009)

Lemmatization is to reduce the inflected words by replacing with root word. For example, studying and studies are replaced by their base form ‘study’, which can lower the amount of analyzing words and normalize text at the same time.

Sentence boundary detection could extract sentences from given text, dividing by comma character. Consider a text “I like summer in Finland. In the summer, I go to forest picking up berries and mushrooms.” This text is split into two meaningful units from sign of dot for further investigation such as entity extraction.

Named entity recognition (NER) can assign a name to some certain words. For example, $10 million is entitled as money in NER feature. Since NER highly depends on pretrained examples of selected model, spaCy is not always promising for NER, with tuning and optimization needed.

Entity detection can recognize critical elements, extracting information effectively from text. With supported entity types provided by spaCy, some words are entity identified and labeled. For instance, numbers are entitled as cardinal, locations are entitled as GPE, etc. (Figure 10)

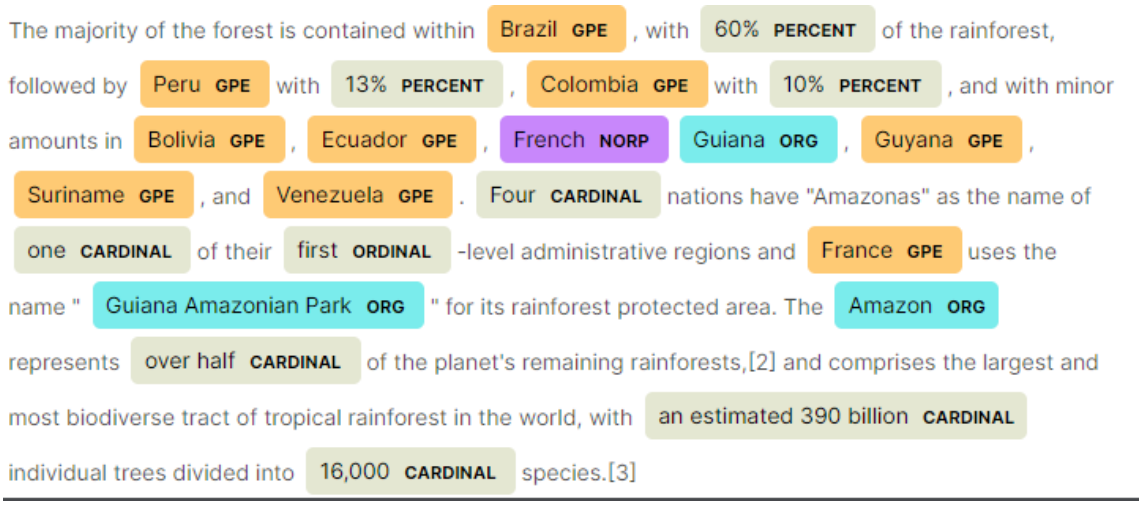


Figure 10. Entity Detection by spaCy

Similarity can be determined by comparing built-in vectors provided by spaCy model. The larger statistical models are loaded, the more vectors are included to compare with. Similarity prediction is a common method to build a recommendation system, and spaCy becomes one possible tool for likeliness prediction among objects.

In this research, “fi_core_news_lg”, a finnish pipeline optimized for CPU, has been chosen for retrieving the embedded vectors of text.

3.3.4 Sentence Transformers

Transformer introduced in 2017 (A.Vashwani) is one of the most powerful and state-to-the-art approaches to NLP tasks using the attention mechanism solving the bottleneck issue between encoder and decoder models that old recurrent neural networks (RNN) created.

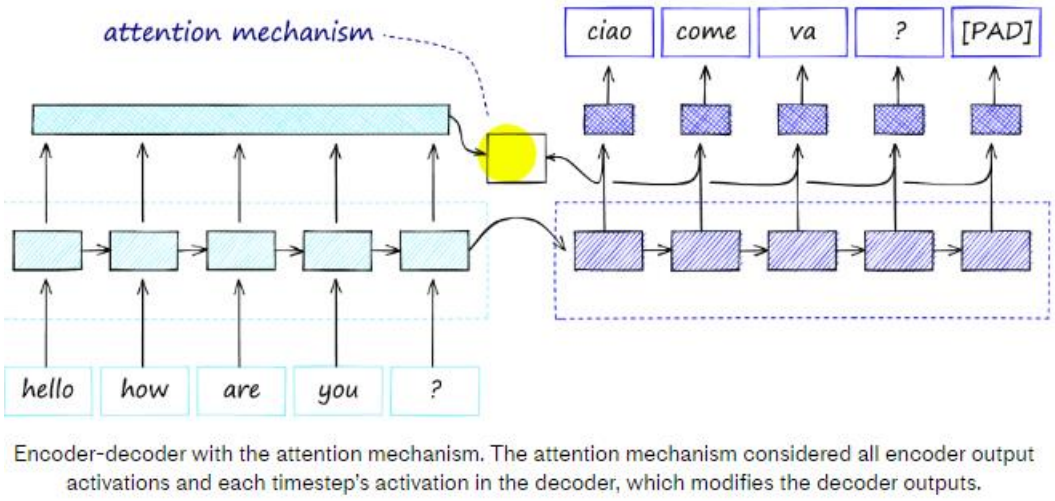


Figure 11. Encoder-decoder with the attention mechanism (PINECORE)

Context vector produced by encoder model is passed into attention mechanism with less information loss. Decoder produces translated text as a continuation of encoder. The attention mechanism can figure out the relationships between all words in a sentence. (Figure 11) For example, given the sentence “The dog’s bark was so vicious”, the transformer pays attention to the word “dog” and determines that the word “bark” refers to the sound made by a dog instead of the outer layer of a tree in this sentence. Transformer contributes a remarkable change to NLP ecosystem with excellent performance and incredible capability for generalization.

Pretrained transformer model achieves various functions simply by shifting the order of last few layers, however, sentence-level embeddings are not the ones transformer model

working on for producing sentence vector, resulting into insufficiently capturing the complete semantic meaning of phrases. Sentence transformers based on SBERT utilizing mean pooling on the final output layer could generate sentence embeddings.

A python library called sentence-transformers provides multiple models for different tasks, such as feature extraction and sentence similarity. “paraphrase-multilingual-MiniLM-L12-v2” is a quick sentence transform model with high quality, mapping sentences and paragraphs to a 384-dimensional dense vector space. (sentence-transformers/paraphrase-multilingual-MiniLM-L6-v2, ei pvm) Pretrained Finnish sentence embeddings are obtained with sentence transformers of “sbert-cased-finnish-paraphrase” trained by Turku NLP Group. For the purpose of this research, those two models have been applied on computing items’ similarity.

3.3.5 SBERT

BERT stands for Bidirectional Encoder Representations from Transformers, delivering an incredible result on implementing NLP tasks. The base of BERT is the transformer architecture with the additional ability of bidirectionally training by understanding the text from left-side and right-side simultaneously. Unlike transformer, BERT as a stack of encoders applies Masked Language Modelling (MLM) by the means of masking 15% words randomly and predicting the masked word within the sequence itself according to the full context of the sentence. (Sanagapati, 2020)

There are many BERT models free of charge in Hugging face, an AI community providing state-of-the-art models for various NLP tasks. One reason BERT gets highly promoted is due to the fact that it is straightforward to use for specific tasks, by simply adding a small layer to the core model. For example, to measure sentence similarity with BERT, a classification head fully connects to the top of the Transformer output. (Figure 12)

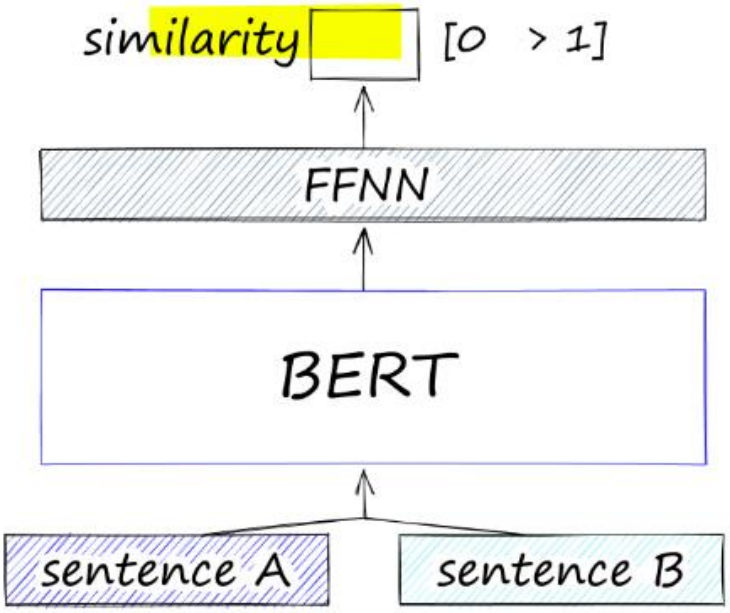


Figure 12. The BERT cross-encoder architecture of a BERT model for sentences similarity (PINECORE)

Compared to cross-encoder architecture, BERT with a Siamese architecture (SBERT) employs mean pooling on the final output layer to acquire a sentence embedding instead of classification head. A Siamese architecture applies on BERT with two identical BERTs in parallel and shares the same network weights. (PINECORE) (Figure 13)

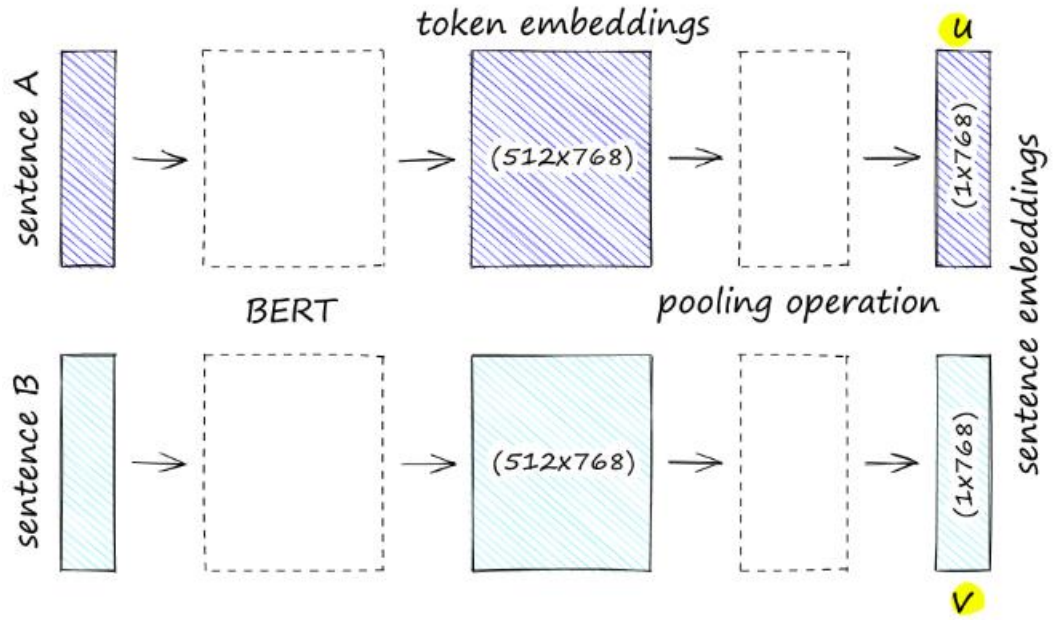


Figure 13. SBERT model for sentence embeddings (PINECORE)

This research has employed two sentence-transformer models from Hugging Face, "paraphrase-multilingual-MiniLM-L12-v2" and "sbert-cased-finnish-paraphrase", to build up recommendation system.

3.4 Evaluation Metrics

Normally, it is not easy to measure the performance of a recommendation system because of insufficient data available to a new business or working on unlabelled data. However, being able to evaluate it in a quantity way is necessary for model selection and optimization. In general, evaluating a recommendation system can be based on well-defined metrics and human judgement. Since the dataset in this research was extracted from a production control system (MES), the performance of recommendation system was incapable to be measured by real users in an offline situation. Therefore, only metrics-based evaluation is illustrated in the paper, leaving human evaluation to expert or future A/B online testing once the recommendation system is embedded into MES.

To apply classification metrics for further evaluation, the dataset should be labelled with a target variable for relevant or irrelevant identification. Moreover, the list retrieved by the recommendation system should be also considered as order-aware or order-unaware. For order-aware system, ranking-based metrics give a better insight on order influence. While for order-unaware system, using decision support metrics ignores the ranks of the suggested items, just focusing on the numbers of relevant items retrieved.

The ranking-focused metric is more useful for a practical system since users pay more attention on top recommendations instead of all the items in the list. In order to understand ranking-based metrics better, decision support metrics are firstly introduced.

3.4.1 Decision support metrics

In a binary classifier, precision and recall are classical metrics to quantitively categorize items into correctly predicted and incorrectly predicted. Expanding to recommendation system, both top k items suggested and items unsuggested are grouped by relevant or not, converting the problem into a binary issue. (Table 1)

Table 1. Confusion matrix of recommendation results (Deutschman, 2023)

	Relevant	Not relevant
Recommended top k	True Positive (TP)	False Positive (FP)
Not recommended	False Negative (FN)	True Negative (TN)

The list including all recommended items can be trimmed into subsets indexed by k considering the first k items. Precision @K and Recall @K are the score of precision and recall calculated for the subsets from rank 1 to k.

$$Precision\ @k = \frac{\#of\ relevant\ recommendations\ (TP)}{\#\ of\ recommended\ items\ (k)(TP + FP)}$$

$$Recall\ @k = \frac{\#\ of\ relevant\ recommendations\ (TP)}{\#\ of\ all\ the\ possible\ relevant\ items\ (TP + FN)}$$

Table 2. Calculation of precision @k (Deutschman, 2023)

P(k=3) = 1/3

Rank	Product	Is recom.	Result
1	product B	1	TP
2	product A	1	FP
3	product E	1	FP
4	product C	1	TP
5	product D	1	TP
6	product G	0	FN
7	product I	0	TN
8	product H	0	TN
9	product F	0	FN

Rank	Product	Is recom.	Result
1	product B	1	TP
2	product A	1	FP
3	product E	1	FP
4	product C	1	TP
5	product D	1	TP
6	product G	0	FN
7	product I	0	TN
8	product H	0	TN
9	product F	0	FN

P(k=5) = 3/5

Table 3. Calculation of recall @k (Deutschman, 2023)

R(k=3)=1/5

Rank	Product	Is recom.	Result
1	product B	1	TP
2	product A	1	FP
3	product E	1	FP
4	product C	1	TP
5	product D	1	TP
6	product G	0	FN
7	product I	0	TN
8	product H	0	TN
9	product F	0	FN

Rank	Product	Is recom.	Result
1	product B	1	TP
2	product A	1	FP
3	product E	1	FP
4	product C	1	TP
5	product D	1	TP
6	product G	0	FN
7	product I	0	TN
8	product H	0	TN
9	product F	0	FN

R(k=5)=3/5

Recall @K is easy to interpret with considering all the relevant items in the whole dataset, however, it can return a deceptive result with a perfect score as increasing k to total number of items.

F1 @K is defined as the harmonic mean of Precision @k and Recall @k, combining those two metrics into one.

$$F1@k = 2 * \frac{precision\ @k * recall\ @k}{precison\ @k + recall\ @k}$$

From the formular above, F1@K gives equal weight to Precision @K and Recall @K, which is designed well for imbalanced dataset.

Precision @K, Recall @K and F1 @K are generally calculated for single item recommendation.

3.4.2 Ranking-based metrics

When a recommendation system returns a descending ordering of items, ranking-based metrics should be considered.

Unlike Precision @K (P@K) which simply takes account the number of relevant items in the subset, Average Precision @K (AP@K) rewards the correct retrieved recommendations on the top of the list.

$$AP\ @k = \frac{1}{m} \sum_{k=1}^{N} Precision\ @k * rel(k)$$

$$rel(k) = \begin{cases} 0, if\ item\ at\ k_{th}\ rank\ is\ relevant \\ 1, \quad otherwise \end{cases}$$

AP@K is the sum of P@K where the item at the k_{th} rank is relevant divided by the total number of relevant items in the top k recommendation list. (Rink, 2023)

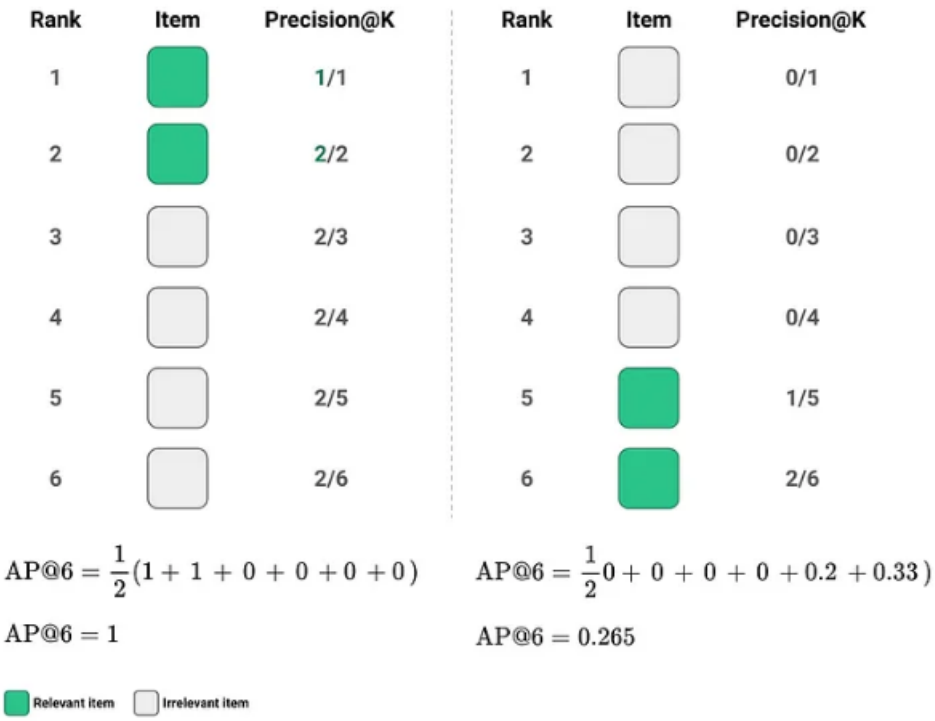


Figure 14. Example of relevant items on different ranks (Rink, 2023)

From figure 14, the values of AP@K are different even with the same number of relevant items, simply due to the ranking difference. AP@6 having relevant items on the first two places in the list obtain higher AP@K score compared to placing them on the bottom of the list. AP@K penalizes recommendations for lower relevant ranks, but it is not able to penalize for including additional irrelevant items into the list. Thus, the model should aim to retrieve the most relevant items and place them on the very first orders. AP@K is typically computed for one item from all recommendations by averaging Precision @1 to Precision @K.

Mean Average Precision @K (MAP@K) implies the mean value of AP@K for all items. (Rink, 2023)(Figure 15)

$$MAP@K = \frac{1}{M}\sum_{j=1}^{M} AP@K$$

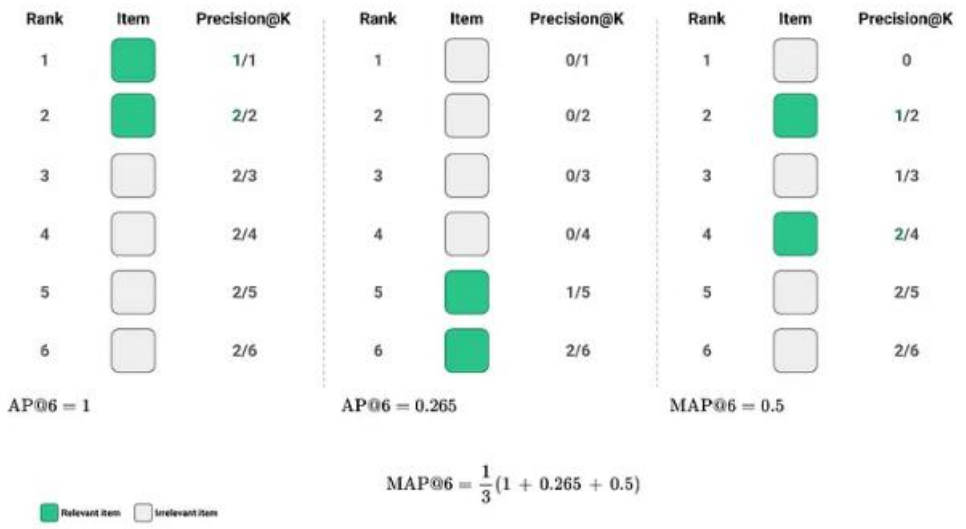


Figure 15. Example of MAP@K (Rink, 2023)

MAP@K is a popular evaluation metric for information retrieval applications, considering returned items as order-aware list. There are other ranking-based metrics which are useful for collaborative-filter recommendation system, such as Normalized Discounted Cumulative Gain (NDCG@K) and Average Reciprocal Hit Rank (ARHR). Since this research only concentrates on building a content-based recommendation system, other ranking-based metrics will not be further explained.

3.4.3 Other metrics

Recommendation does not mean prediction, thus other metrics besides of accuracy related should be studied.

Coverage, one of recommendation-centric metrics, represents the percentage of recommended items in the whole dataset. The result can be close to 100%, proving that the system can suggest every single item. However, for the popularity-focused system, coverage value is nearly 0% by only retrieving top k out of all items. (Deutschman, 2023)
From the perspective of business, metrics for cold-start problem are not sufficient because of lacking the reaction and feedback from real users. A/B testing is a reliable

method, statistically measuring the recommendation impact on business. (Deutschman, 2023)

4 Case Study

A manufacturing company has operated on a production control system (MES) to record daily work activity. Employees have reported production issues into system for further investigation. With the amount of issue cases increasing, the owner of the system was curious about the possibility to utilize history data for predicting some attributes of new production issue. The preliminary idea was to build a recommendation system focusing on issue description text. If the result satisfies, the recommending function might be added to new version of MES.

4.1 Data collection and introduction

The data from production control system (MES) was recorded and managed by around 60 different people and it could be extracted and saved in an excel csv file. This research only focused on the data released from 17/11/2016 to 10/06/2022. The rows standing for cases of the production issue were created by factory employees from their daily working activity. The columns represent different attributes for each case. Read from python, the data contains 30,751 instances and 65 features. Luckily, not all the variables match the interest for a recommendation system, meaning that the data could be reduced in terms of feature selection.

"COMMENT_TEXT" is the key variable referring to the issue description with a human free-text in a digital style. One example sentence from "COMMENT_TEXT" is like "Sumpun putkien päiden suojaus toteutettu monella erilaisella tulpalla ja 3 eri väriä. tulpat pieniä ja jää putken sisään ja niitä ei meinaa huomata. (allas oli myös tosi likainen vaikka suojattu) Voisiko suojatulpat muuttaa näkyvämmäksi ja isommalla ulostulevalla kauluksella?" To track cases easily, "DEVIATION_NO" has been chosen due to its unique values for every single case. A feature should be selected as a target variable for performance evaluation, such as a categorical variable called "REASON_CLASS". "DEVIATION_REASON_CODE_ID" and "ROOTCAUSE_REASON_CODE_ID" are two subclass variables out from "REASON_CLASS", which are also included in the subset on the purpose

of further display of “REASON_CLASS” in a recommendation system. The last chosen variable is “MATERIAL_DESCRIPTION” due to the interest of correlation between description of material and production issue. After feature selection and duplicates removing, the data is in the shape of 30,748 instances and 6 features.

The unique number for those 6 variables is checked, especially on those potential categorical variables for performance evaluation. There are 12 classes for “REASON_CLASS”, 72 for “DEVIATION_REASON_CODE_ID”, and 259 for “ROOTCAUSE_REASON_CODE_ID”, inferring that employees have many options to choose category and misclassification might easily take place as large number of subclasses available. Therefore, it is better to use “REASON_CLASS” with a smaller number of classes for measuring the accuracy of recommendation system.

It is not necessary to analyse missing values in the dataset, which can be handled by deleting them. One thumb of rule is to drop entire column if this column has more than half of the rows as null. Therefore, “MATERIAL_DESCRIPTION” is removed due to containing 16,341 missing values. There is no point to put effort on analysing “NaN” values from “COMMENT_TEXT”, so 6,692 instances are also removed. The data is finally presented with 24,056 instances and 5 features left.

4.2 Data Pre-processing

The raw text recorded by different people can be ambiguity and inconsistent. Text pre-processing is an essential step to clean up the sentences from key variables, “COMMENT_TEXT” in this thesis. There are some common text pre-processing steps, such as lowercasing, removing stopwords, removing punctuations, tokenization, stemming, lemmatization, etc.

Imported NLTK library, several packages for text pre-processing can be downloaded. All the words in the sentences are lowercased, which is helpful for TF-IDF, avoiding token duplication with different casings.

Non-alphanumeric character and punctuation using regular expression patterns (A.M.Kuchling, ei pvm) are removed. The length of sentences is limited between 4 words and 200 words and sentences out of such limit range are dropped as well.

Stopwords are frequent words with less valuable information, which are always removed from the corpus. Different libraries like spaCy, gensim, Scikit-learn, contain different amount of stopwords. In this research, 179 stopwords from NLTK library are detected and removed. (Figure 16)

{'olla', 'en', 'nyt', 'tänä', 'mihin', 'ovat', 'meille', 'tässä', 'poikki', 'et', 'heidät', 'kuka', 'teidät', 'ei', 'keille',
'olisit', 'minun', 'siitä', 'sen', 'emme', 'näitä', 'joina', 'heistä', 'olen', 'olin', 'hänen', 'oli', 'tästä', 'minut', 'siin
ä', 'jota', 'meihin', 'noille', 'mistä', 'kenessä', 'eivät', 'minua', 'niiksi', 'sinua', 'tuona', 'keissä', 'sinulle', 'hänet',
'olitte', 'tallä', 'minuun', 'teidän', 'niissä', 'keinä', 'mitkä', 'se', 'olisitte', 'sitä', 'keiksi', 'vaan', 'mikä', 'sinun',
'miltä', 'joksi', 'sinulla', 'tuon', 'minä', 'olit', 'noiden', 'noihin', 'tuolla', 'heihin', 'tätä', 'jotka', 'olisivat', 'meis
tä', 'jossa', 'noilta', 'kuin', 'olette', 'häneltä', 'meissä', 'tälle', 'tuossa', 'näinä', 'kenellä', 'olet', 'heille', 'noist
a', 'niitä', 'minulta', 'näillä', 'kenen', 'joka', 'keiden', 'jonka', 'te', 'tuolle', 'joilta', 'sinulta', 'noilla', 'keitä',
'olleet', 'hänessä', 'tuo', 'kenestä', 'teitä', 'tuotä', 'kenenä', 'minkä', 'itse', 'sinä', 'mutta', 'niin', 'näihin', 'meidä
n', 'mukaan', 'häntä', 'keillä', 'heillä', 'sinuun', 'hänellä', 'ketkä', 'mitä', 'joihin', 'koska', 'noissa', 'meillä', 'näist
ä', 'johon', 'tuolta', 'niille', 'että', 'millä', 'ne', 'hänelle', 'niiltä', 'tämä', 'näille', 'sillä', 'noita', 'noin', 'kanss
a', 'tai', 'keihin', 'kenet', 'heiltä', 'sekä', 'siltä', 'olisin', 'teihin', 'joista', 'vaikka', 'yli', 'keneltä', 'ole', 'heis
sä', 'minulla', 'heitä', 'nuo', 'joilla', 'olemme', 'sinusta', 'tuoksi', 'joiden', 'joita', 'olivat', 'heidän', 'sille', 'joiss
a', 'me', 'minusta', 'teiltä', 'minulle', 'sinussa', 'jolla', 'hänestä', 'tuosta', 'olimme', 'nämä', 'missä', 'tuohon', 'miks
i', 'joiksi', 'tältä', 'jona', 'teissä', 'hän', 'jolle', 'olisi', 'niistä', 'näiltä', 'ja', 'teille', 'näissä', 'näiden', 'sinu
t', 'keneksi', 'teistä', 'on', 'siksi', 'he', 'joille', 'jos', 'ketä', 'tämän', 'siihen', 'keiltä', 'näiksi', 'mille', 'vai',
'häneen', 'niihin', 'niinä', 'noina', 'kun', 'josta', 'meiltä', 'noiksi', 'keistä', 'niiden', 'niillä', 'ette', 'minussa', 'jol
ta', 'keneen', 'olisimme', 'ollut', 'teillä', 'tähän', 'meidät', 'täksi', 'kenelle', 'meitä'}

Figure 16. Finnish Stopwords with NLTK library

Tokenization is the process transforming a text into a list of sentences, words or even characters called tokens. The sentences from “COMMENT_TEXT” are split into words. Lemmatization is to convert the word to a generic form, such as inflected words of “running, ran, runs” to generic word of “run”. Stemming reduced the words to their root forms, but the root word might not be a valid one compared with lemmatization. Both ways of changing words are durable. Voikko, a linguistic tool works effectively on Finnish lemmatization. However, installing and implementing it in Windows system is complicated. Thanks that stemming with “SnowballStemmer” performs good enough in Finnish language. Therefore, stemming is done for text pre-processing instead of lemmatization.

After all the pre-processing mentioned above, three new columns are added to the dataset. They are “clean_COMMENT_TEXT” that the sentence is mainly alphanumerical text, “tok_stem_COMMENT_TEXT” with lists of stemming and tokenized data, and “clean_tok_stem_COMMENT_TEXT” that combines tokens into new sentences.

4.3 Data Analysis and Visualization

Data visualization is a trivial strategy in exploratory data analysis, especially for big data. People can quickly understand the trend of data, detect abnormalities, and make general summarizes from data graphics, such as charts, plots, histograms, etc.

The distribution of “REASON_CLASS” is visualized as histogram. (Figure 17) There are big differences among 12 reason classes with some classes like “Tuotteen laaduttomuus” containing up to 8811 instances and some classes like “Automatic Deviation For Missing Part” having less than 10 instances. The data is apparently imbalanced in terms of classification. However, the recommendation system is built solely based on “COMMENT_TEXT” without any target variable involved for model training. “REASON_CLASS” works simply as the role of evaluation reference variable. In this case, it is not necessary to balance dataset since the Mean Average Precision @K (MAP@K) are calculated for every single instance. Even though by removing some minor classes could slightly increase the MAP@K score, (Figure 18) it does not make sense to intentionally delete that information with the potential risks of more instances adding to minor classes in MES later on.

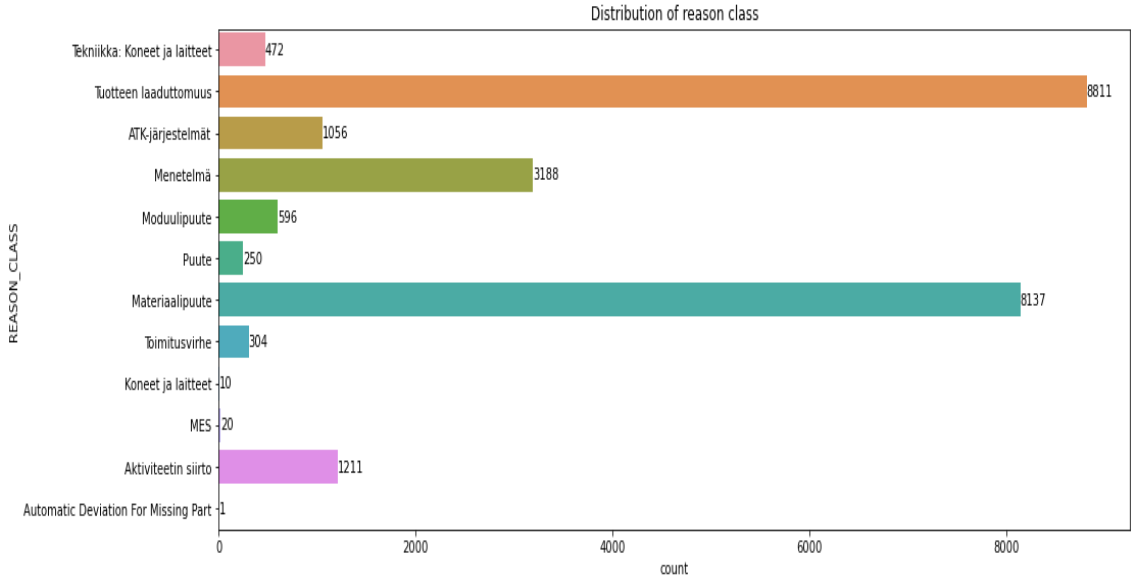


Figure 17. Distribution of 12 reason classes

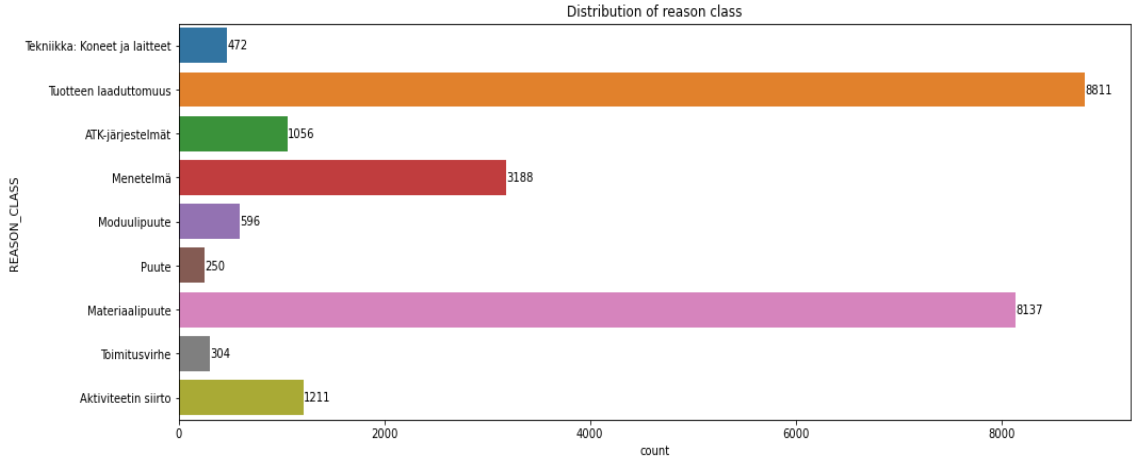


Figure 18. Distribution of 9 reason classes after removing minor ones

Wordclouds is a common example of quantities visualization, representing word frequency. The larger the words are, the more often the words are appeared in the texts. The figures of wordclouds for every reason class are shown in Appendix 1. (Appendix 1)

N-grams analysis is a straightforward method for text mining by analysing the neighbouring sequences of tokens in a document. N is a positive integer, standing for the number of words in a sequence. For example, in the sentence “The girl walks through the corridor.” When N=1 (known as unigram), the N-grams is “the”, “girl” “walks” “through”, “the”,

“corridor”; when N=2 (known as bigram), the N-grams is “the girl”, “girl walks”, “walks through”, “through the”, “the corridor”; when N=3 (known as trigram”), the N-grams is “the girl walks”, “girl walks through”, “walks through the”, “through the corridor”. In this research, N-grams are developed as unigram, bigrams, and trigrams for production issue descriptions. From figures below, bigrams and trigrams are effective for extracting issue information. (Figure 19) (Figure 20) (Figure 21) “Puutu” is the most common word derived from different N-grams models. This makes sense in a way because “Puutu” is “missing” related and missing can be covered over a couple of reasons, such as missing instruction (ohje puuttuu), missing part (osa puuttuu), missing from shelf (puuttuu hyllystä), missing from collection (puuttuu keräyksestä). (Kangas, 2021) More useful information can be read by listing frequent words, such as “ei voi asent”, “ei ole”, “ei sovi paika”. It is normal with many occurrence of negative words due to the “COMMENT_TEXT” standing for the issue taken place in the production line.

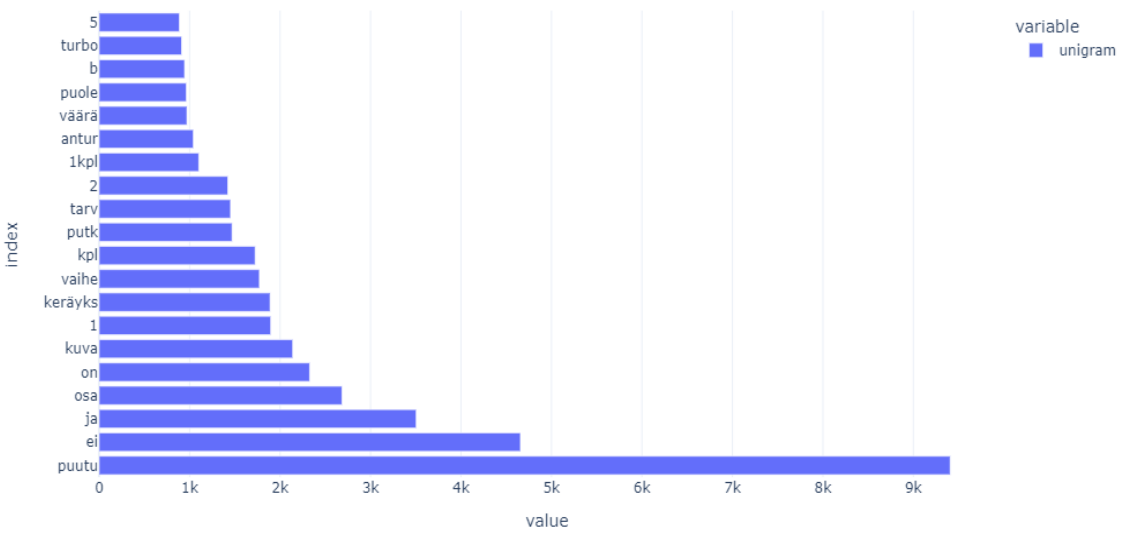


Figure 19. Unigram of cleaned “COMMENT_TEXT” sentence

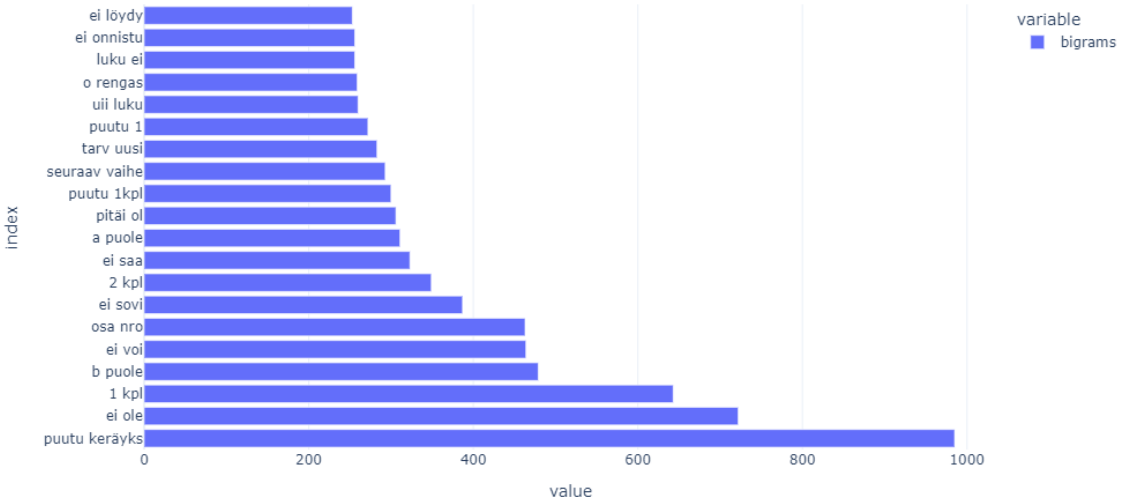


Figure 20. Bigrams of cleaned "COMMENT_TEXT" sentence

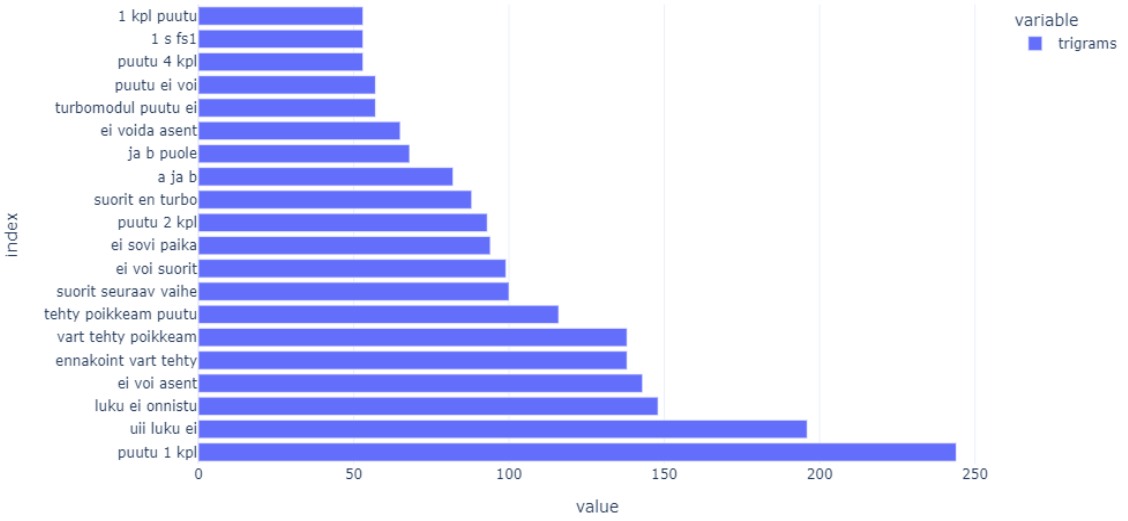


Figure 21. Trigrams of cleaned "COMMENT_TEXT" sentence

4.4 Model Training and Recommendation

The recommendation system has used sentences from "COMMENT_TEXT" and applied NLP-based machine learning model to represent each description text as a numerical

feature vector with the size of 21,117. An embedded matrix generated from those feature vectors covers all the information of product issue description in the shape of (24,056, 21,117). Each row vector from embedded matrix stands for one issue case, that new query sentence vector can be computed with, to get a cosine distance. (Figure 22) This research has experimented with TF-IDF, Word2Vec, spaCy, sentence transformers and SBERT, to build recommendation system.

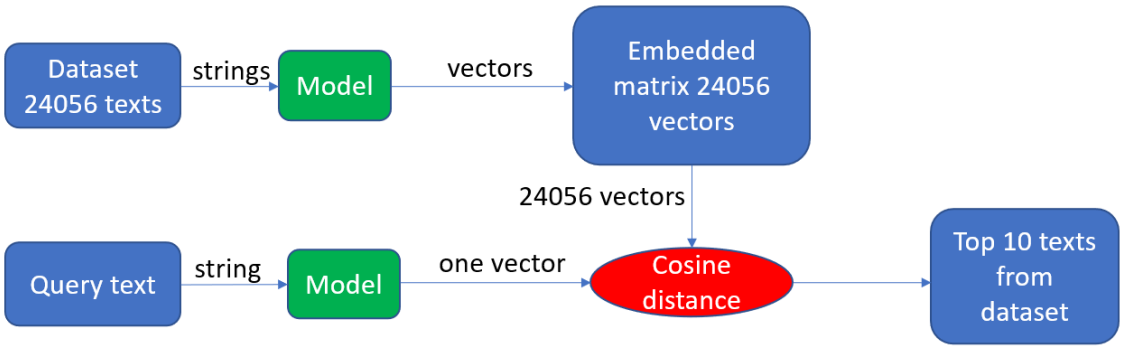


Figure 22. Process of building recommendation system (LD, 2021)

4.4.1 TF-IDF

The first model has used TF-IDF to measure similarity between query text and issue descriptions for the whole dataset. After fitting “clean_tok_stem_COMMENT_TEXT” to the TF-IDF model, an embedding vector with 21,117 dimensions is generated for each issue description. A feature matrix with the shape of (24,056, 21,117) contains the issue description information of the whole dataset. The recommendation system retrieves the first 10 instances in the descending order of average cosine distance between query text vector and each row vector in embedded feature matrix. TD-IDF model is simple to understand and fast to train.

Let’s take an issue description to suggest relevant issue cases with TF-IDF-based model. The query sentence is “sentryläpiviennit puuttu keräyksestä” randomly chosen from “COMMENT_TEXT”, and the top 10 recommendations are shown as below. (Table 4)

Table 4. Display of Recommendation system based on TF-IDF model

DEVIATION_NO	REASON_CLASS	DEVIATION_REASON_CODE_ID	ROOTCAUSE_REASON_CODE_ID	COMMENT_TEXT
QD0171050	Materiaalipuute	1.1. KOMP_PUUTE	1.1.1. SALDOMAT	puuttuu keräyksestä
QD0125019	Materiaalipuute	1.1. KOMP_PUUTE	1.1.1.4. KOTKUTSU	puutuu keräyksestä
QD0071883	Materiaalipuute	1.1. KOMP_PUUTE	1.1.1.1. SALDOHEITTO	keräyksestä puuttuu
QD0178153	Menetelmä	5.4. MUU	5.4.1. MUU	Puuttuu keräyksestä!
QD0204973	Materiaalipuute	1.1. KOMP_PUUTE	1.1.1. SALDOMAT	Puuttuu keräyksestä.
QD0174560	Menetelmä	5.6. TOIMINTATAPAVIR	5.6.1. OPAST_PUUTTEE	Puuttuu keräyksestä.
QD0170284	Menetelmä	5.6. TOIMINTATAPAVIR	5.6.1. OPAST_PUUTTEE	puuttuu keräyksestä
QD0181382	Materiaalipuute	1.4. Ostopuute	1.4.8.Delivery Late	puuttuu keräyksestä
QD0111862	Materiaalipuute	1.1. KOMP_PUUTE	1.1.1. SALDOMAT	puutuu keräyksestä
QD0182923	Menetelmä	5.6. TOIMINTATAPAVIR	5.6.1. OPAST_PUUTTEE	Puuttuu keräyksestä.

If “REASON_CLASS” variable is utilized for calculating quantity evaluation result, Precision @10 (P@10) is 0.60 and Average Precision @10 (AP@10) in TF-IDF-based recommendation system is 0.85.

4.4.2 Word2Vec

Word2Vec is a simple neural network with two layers. The model trains in a Continuous Bag of Words (CBOW) architecture on a few epochs with token list fitted. Word2Vec model is not able to produce vectors for the words that were not in the corpus token list, hence those words in the query document should be removed. Different from other approaches, Word2Vec is worse to process stemming words which are not valid root ones in a corpus. Therefore, only tokenized word lists are fitted in the training model for word embedding extraction.

Word2Vec gives the following recommendations for the query sentence “sentryläpivienit puuttu keräyksestä”. (Table 5)

Table 5. Display of Recommendation system based on Word2Vec model

DEVIATION_NO	REASON_CLASS	DEVIATION_REASON_CODE_ID	ROOTCAUSE_REASON_CODE_ID	COMMENT_TEXT
QD0171967	Materiaalipuute	1.1. KOMP_PUUTE	1.1.1. SALDOMAT	Puuttuu keräyksestä (PAAF307923)
QD0058596	Materiaalipuute	1.1. KOMP_PUUTE	1.1.1. SALDOMAT	PAAC301134 Puuttuu keräyksestä
QD0090503	Materiaalipuute	1.1. KOMP_PUUTE	1.1.1. SALDOMAT	PAAF325132 puuttuu keräyksestä.
QD0127307	Materiaalipuute	1.1. KOMP_PUUTE	1.1.1.3. VÄÄ_MÄÄ_TAR	paac301967 puuttuu keräyksestä
QD0117834	Materiaalipuute	1.1. KOMP_PUUTE	1.1.1. SALDOMAT	PAAF140532 PUUTTUU KERÄYKSESTÄ
QD0163476	Materiaalipuute	1.1. KOMP_PUUTE	1.1.1.3. VÄÄ_MÄÄ_TAR	jatkopala puuttuu keräyksestä
QD0183847	Materiaalipuute	1.1. KOMP_PUUTE	NaN	PAAF261499 Puuttuu keräyksestä.
QD0102910	Materiaalipuute	1.1. KOMP_PUUTE	1.1.1. SALDOMAT	PUUTTUU keräyksestä\n\npaac301098
QD0072820	Materiaalipuute	1.1. KOMP_PUUTE	NaN	puuttuu keräyksestä paaf514936
QD0058588	Materiaalipuute	1.1. KOMP_PUUTE	NaN	PAAF108409 puuttuu keräyksestä

P@10 and AP@10 are 1 by suggesting the top10 relevant issue cases all belonging to correct reason class.

4.4.3 SpaCy

SpaCy contains multiple trained pipelines for various languages, including the processing of tokenization, lemmatization, and word-level vectorization. By loading a Finnish pipeline optimized for CPU, it is adequate simply providing sentences from “COMMENT_TEXT” without pre-processing. Pre-trained weights from “fi_core_web_lg” pipeline contains 200k floret vectors in 300 dimensions trained on Finnish webpages corpus.

With the same query sentence “sentryläpiviennit puuttu keräyksestä”, spaCy suggests the following instances. (Table 6)

Table 6. Display of Recommendation system based on spaCy model

DEVIATION_NO	REASON_CLASS	DEVIATION_REASON_CODE_ID	ROOTCAUSE_REASON_CODE_ID	COMMENT_TEXT
QD0171050	Materiaalipuute	1.1. KOMP_PUUTE	1.1.1. SALDOMAT	puuttuu keräyksestä
QD0026566	Puute	4. KERAYSPUUTE	NaN	puuttuu keräyksestä
QD0170284	Menetelmä	5.6. TOIMINTATAPAVIR	5.6.1. OPAST_PUUTTEE	puuttuu keräyksestä
QD0180774	Materiaalipuute	1.1. KOMP_PUUTE	1.1.2. BULK	puuttuu keräyksestä
QD0085064	Materiaalipuute	1.1. KOMP_PUUTE	1.1.1.3. VÄÄ_MÄÄ_TAR	puuttuu keräyksestä
QD0082948	Materiaalipuute	1.1. KOMP_PUUTE	1.1.1. SALDOMAT	puuttuu keräyksestä
QD0128025	Materiaalipuute	1.1. KOMP_PUUTE	1.1.1. SALDOMAT	puuttuu keräyksestä
QD0094989	Materiaalipuute	1.1. KOMP_PUUTE	1.1.1. SALDOMAT	puuttuu keräyksestä
QD0048411	Materiaalipuute	1.1. KOMP_PUUTE	1.1.1. SALDOMAT	puuttuu keräyksestä
QD0048748	Materiaalipuute	1.1. KOMP_PUUTE	1.1.1. SALDOMAT	puuttuu keräyksestä

Even though spaCy is expected to be more powerful than TF-IDF, for this sentence both P@10 and AP@10 are lower than TD-IDF with 0.80 and 0.73 respectively.

4.4.4 Sentence Transformers

Sentence Transformers provide high-level pretrained models, achieving more robust and accurate result. Two models are tested to find semantically similar sentences within one language or across languages. They are “paraphrase-multilingual-MiniLM-L12-v2” trained on parallel data for over 50 languages and “TurkuNLP/sbert-cased-finnish-paraphrase” mainly focus on Finnish.

Query sentence like “sentryläpiviennit puuttu keräyksestä” obtains following top 10 recommendations. (Table 7) (Table 8)

Table 7. Display of Recommendation system based on Turku-NLP transformers model

DEVIATION_NO	REASON_CLASS	DEVIATION_REASON_CODE_ID	ROOTCAUSE_REASON_CODE_ID	COMMENT_TEXT
QD0190767	Materiaalipuute	1.1. KOMP_PUUTE	1.1.3. KANBAN	kannake puuttuu keräyksestä
QD0185245	Materiaalipuute	1.1. KOMP_PUUTE	1.1.1.3. VÄÄ_MÄÄ_TAR	Puuttuu läpivienti keräyksestä
QD0177350	Materiaalipuute	1.1. KOMP_PUUTE	1.1.1. SALDOMAT	puuttuu keräyksestä.
QD0179880	Materiaalipuute	1.1. KOMP_PUUTE	1.1.1.3. VÄÄ_MÄÄ_TAR	puuttuu keräyksestä.\n
QD0176975	Materiaalipuute	1.4. Ostopuute	1.4.6.Inacc.stock	puuttuu keräyksestä.
QD0202176	Materiaalipuute	1.4. Ostopuute	1.4.3. Defective mat	puuttuu keräyksestä.
QD0172336	Materiaalipuute	1.4. Ostopuute	1.4.5.New reservatio	puuttuu keräyksestä.
QD0180253	Materiaalipuute	1.1. KOMP_PUUTE	1.1.1.3. VÄÄ_MÄÄ_TAR	puuttuu keräyksestä.
QD0182892	Materiaalipuute	1.1. KOMP_PUUTE	NaN	puuttuu keräyksestä.
QD0179987	Menetelmä	5.3. SETITYSVIRHE	5.3.1. SETITYS	puuttuu keräyksestä.

Table 8. Display of Recommendation system based on multilingual transformers model

DEVIATION_NO	REASON_CLASS	DEVIATION_REASON_CODE_ID	ROOTCAUSE_REASON_CODE_ID	COMMENT_TEXT
QD0027996	Puute	4. KERAYSPUUTE	4.2 PUUT_KERAYS	sentryjen keräys puuttuu
QD0082297	Materiaalipuute	1.1. KOMP_PUUTE	1.1.1. SALDOMAT	Putki puuttuu keräyksestä
QD0197424	Materiaalipuute	1.1. KOMP_PUUTE	NaN	Putki puuttuu keräyksestä
QD0145641	Materiaalipuute	1.4. Ostopuute	1.4.8.Delivery Late	Putki puuttuu keräyksestä
QD0205916	Materiaalipuute	1.1. KOMP_PUUTE	1.1.2. BULK	O rengas puuttuu keräyksestä
QD0060362	Materiaalipuute	1.1. KOMP_PUUTE	1.1.1.1. SALDOHEITTO	putki puuttuu keräyksestä
QD0109475	Materiaalipuute	1.1. KOMP_PUUTE	NaN	putki puuttuu keräyksestä
QD0196581	Menetelmä	5.6. TOIMINTATAPAVIR	5.6.1. OPAST_PUUTTEE	putki puuttuu keräyksestä
QD0044116	Puute	4. KERAYSPUUTE	4.2 PUUT_KERAYS	Holkit puuttuu keräyksestä
QD0192285	Materiaalipuute	1.1. KOMP_PUUTE	1.1.2. BULK	puuttuu hyllystä ja keräyksestä

There is an obvious improvement employing sentence transformers-based models, especially on Turku-NLP sentence transformers. P@10 and AP@10 are 0.90 and 1.00 using Turku-NLP pretrained models. And P@10 and AP@10 are 0.70 and 0.73 using multilingual pretrained models. Compared to the previous methods, the result of this query sentence with Turku-NLP transformers model is very good mainly due to the reason that the most relevant items are ranked on the top of the list. However, encoding sentences for corpus embeddings with sentence transformers cost longer time, especially Turku-NLP transformers model with 1h 23 min CPU time.

4.4.5 SBERT

Sentence transformers with Hugging Face architecture add mean pooling on the final output layer to infer a vector for each word and average them for each sentence. For avoiding loading out of memory, the model generates embed matrix with a batch of data. Compared to the implementation with sentence transformers, the code for SBERT is longer by customizing an additional class bundling a series of functions, such as tokenization with AutoTokenizer, loading pretrained model with AutoModel, creating embedded matrix, adding mean_pooling layer, etc. SBERT seems to be more complex but also flexible for self-design model structure available.

The SBERT-based recommendation system retrieves the first 10 relevant items for the same query test sentence “sentryläpiviennit puuttu keräyksestä “as below. (Table 9) (Table 10)

Table 9. Display of Recommendation system based on Turku-NLP SBERT model

DEVIATION_NO	REASON_CLASS	DEVIATION_REASON_CODE_ID	ROOTCAUSE_REASON_CODE_ID	COMMENT_TEXT
QD0190767	Materiaalipuute	1.1. KOMP_PUUTE	1.1.3. KANBAN	kannake puuttuu keräyksestä
QD0185245	Materiaalipuute	1.1. KOMP_PUUTE	1.1.1.3. VÄÄ_MÄÄ_TAR	Puuttuu läpivienti keräyksestä
QD0160056	Materiaalipuute	1.3. KOK_KERÄYS_PUUT	1.3.1. KER_MYÖHÄSSÄ	puuttuu keräyksestä.
QD0050832	Materiaalipuute	1.1. KOMP_PUUTE	1.1.1.1. SALDOHEITTO	puuttuu keräyksestä.
QD0159567	Materiaalipuute	1.3. KOK_KERÄYS_PUUT	1.3.3. KER_NOST_VAIH	puuttuu keräyksestä.\n
QD0179987	Menetelmä	5.3. SETITYSVIRHE	5.3.1. SETITYS	puuttuu keräyksestä.
QD0202176	Materiaalipuute	1.4. Ostopuute	1.4.3. Defective mat	puuttuu keräyksestä.
QD0183976	Materiaalipuute	1.1. KOMP_PUUTE	1.1.1. SALDOMAT	puuttuu keräyksestä.
QD0176282	Materiaalipuute	1.1. KOMP_PUUTE	1.1.1. SALDOMAT	puuttuu keräyksestä.
QD0201803	Materiaalipuute	1.4. Ostopuute	1.4.7 Supplier conf.	puuttuu keräyksestä.

Table 10. Display of Recommendation system based on multilingual SBERT model

DEVIATION_NO	REASON_CLASS	DEVIATION_REASON_CODE_ID	ROOTCAUSE_REASON_CODE_ID	COMMENT_TEXT
QD0027996	Puute	4. KERAYSPUUTE	4.2 PUUT_KERAYS	sentryjen keräys puuttuu
QD0145641	Materiaalipuute	1.4. Ostopuute	1.4.8.Delivery Late	Putki puuttuu keräyksestä
QD0197424	Materiaalipuute	1.1. KOMP_PUUTE	NaN	Putki puuttuu keräyksestä
QD0082297	Materiaalipuute	1.1. KOMP_PUUTE	1.1.1. SALDOMAT	Putki puuttuu keräyksestä
QD0205916	Materiaalipuute	1.1. KOMP_PUUTE	1.1.2. BULK	O rengas puuttuu keräyksestä
QD0196581	Menetelmä	5.6. TOIMINTATAPAVIR	5.6.1. OPAST_PUUTTEE	putki puuttuu keräyksestä
QD0109475	Materiaalipuute	1.1. KOMP_PUUTE	NaN	putki puuttuu keräyksestä
QD0060362	Materiaalipuute	1.1. KOMP_PUUTE	1.1.1.1. SALDOHEITTO	putki puuttuu keräyksestä
QD0044116	Puute	4. KERAYSPUUTE	4.2 PUUT_KERAYS	Holkit puuttuu keräyksestä
QD0192285	Materiaalipuute	1.1. KOMP_PUUTE	1.1.2. BULK	puuttuu hyllystä ja keräyksestä

For Turku-NLP SBERT model, P@10 is 0.90 and AP@10 is 0.95. while for multilingual SBERT model, P@10 and AP@10 are 0.7. Apparently, Turku-NLP SBERT performs better than multilingual model, however, training Turku-NLP model takes 2h 23min CPU times compared with 1h 5min for multilingual one. Normally, running time can be shortened much with the same training configuration if GPU is accessible.

4.5 Results

As for the query sentence “sentryläpiviennit puuttu keräyksestä”, Word2Vec achieves the best result, recommending the first ten relevant issue cases sharing the same reason class as the query sentence. However, we cannot guarantee that Word2Vec always performs the best on other issue cases. To check AP@10 for other production issues, a few query sentences from different reason classes were tested. (Table 11)

Table 11. Average Precision @10 for more test sentences

TF-IDF	Word2Vec	spaCy	Multilingual Sentence Transform-ers	Turku-NLP Sentence Transform-ers	Multilingual SBERT	Turku-NLP SBERT
kiertokangenalaosien vaarnanreiissä ruostetta.						
1	1	1	1	1	1	1
Vaiheensiirto asennetaan kun moottori on siirretty pois asennuspetiltä.						
1	0.5	0.74	0.81	1	0.81	1
linja 2 vaihe 5 rehobot pumppu lähetetty korjattavaksi 4-5 kk sitten ei oo sen jälkeen näkyny						
0	0.26	0.29	0.33	0.33	0.33	0.33
puuttuu keräyksestä						
0.16	0.35	0	0.40	0.29	0	0

From the table above, there are relatively big differences on AP@10 with those NLP methods based on test sentences. “kiertokangenalaosien vaarnanreiissä ruostetta.” achieves perfect result, while “puuttuu keräyksestä” from reason class of “Menetelä” obtains poor AP@10. Word2Vec cannot perform best on all the other test sentences, therefore the metric working on all the production issue cases should be computed to evaluate NLP-based models.

Mean Average @K (MAP@K) can measure the general precision for the whole dataset, taking order importance into consideration. The result is shown in the Table 12.

Table 12. Mean Average Precision @10 for five models

NLP methods	MAP@10
TF-IDF	0.58
Word2Vec	0.56
spaCy	0.60
Multilingual Sentence Transformers	0.64
Turku-NLP Sentence Transformers	0.67
Multilingual SBERT	0.62
Turku-NLP SBERT	0.64

Deep learning models (Sentence Transformers and SBERT) perform better than other models. And Turku-NLP pretrained models achieve slightly better MAP@10 score compared to Multilingual models. The outperformance of deep learning methods implies that the dataset used for this research is relatively large enough for model training with deep learning algorithms. The results obtained from machine learning and deep learning models can be compared with dummy classifier by assigning all samples to the biggest reason class. The precision of dummy classifier is 0.34, indicating that all those methods applied in this research are more reliable.

AP@10 for every single model has been demonstrated by histograms shown in Appendix 2. The distribution of AP@10 proves the outperformance of sentence transformers using Turku-NLP pretrained model, with the highest proportion of forever-correct recommendation (AP@10=1) and the lowest proportion of never-correct recommendation (AP@10=0). Word2Vec provides quite clearly the worst result with almost equal proportion of forever-correct (AP@10=1) and never-correct (AP@10=0) recommendations.

It would be also interesting to investigate MAP@10 for every reason class. (Appendix 3) Seen from the distribution plots, MAP@10 score is highly depended on the number of instances for each class. The larger reason class is, the higher MAP@10 would achieve. Assuming of increasing MAP@10 by including larger reason classes, the last three reason

classes “MES”, “Koneet ja laitteet” and “Automatic Deviation For Missing Part” have been removed as a test. However, there is no further improvements as expected. (Table 13) Therefore, it is not necessary to spare effort on data reduction.

Table 13. Mean Average Precision @10 for five models with three minor reason classes removed

NLP methods	MAP@10
TF-IDF	0.58
Word2Vec	0.56
spaCy	0.60
Multilingual Sentence Transformers	0.64
Turku-NLP Sentence Transformers	0.67
Multilingual SBERT	0.62
Turku-NLP SBERT	0.65

Time cost is another factor that people consider for picking up models. Even though Turku-NLP-based sentence transformer performs the best, the training time takes longer than other deep learning models. (Table 14) Model selection should be according to trade-off strategy case by case.

Table 14. Time consumption on deep learning model training

Model	CPU Times
Multilingual Sentence Transformers	31min 27s
Turku-NLP-based Sentence Transformers	1h 20min 36s
Multilingual SBERT	1h 5min 47s
Turku-NLP-based SBERT	2h 23min 26s

5 Conclusions, Discussions and Future Works

The main purpose of this research is developing a recommendation system providing machine learning NLP-based models that would upon by feeding production issue descriptions, to retrieve relevant history production issue cases, which could potentially add values to the manufacturing process as a new feature. The data was collected from MES and fitted into several models either as the original form or as the converted format, such as token list. Five different NLP-methods have been applied into modelling, which are TD-IDF, Word2Vec, spaCy, Sentence Transformers and SBERT. Multilingual and Turku-NLP pretrained models were utilized for deep learning techniques. Relevant items have been extracted based on the similarity distance between query sentence and issue descriptions of the whole dataset. The quality of recommendation system was obtained through the usage of evaluation metrics, such as Precision @10 (P@10), Average Precision @10 (AP@10) and Mean Average Precision @10 (MAP@10). The highest MAP@10 score was 0.67 with Turku-NLP-based sentence transformer model on the basis of experimenting reason class as target variable to compute with. Applying the best model on MES might save much time and resources, avoiding solving the manufacturing problem from the very beginning without any reference. As the information retrieved are listed in a similarity descending order, people might easily draw some conclusions by simply looking into top recommendations instead of reading through all the items in the list.

The scope of the research covers data collection, data pre-processing, exploratory data analysis, model training, model testing and performance evaluation. The user-interface of recommendation system in MES is not included in the research task.

Previous research analyzing on similar dataset was done by Mrs. Kangas, to categorize issue descriptions into different deviation groups. (Kangas, 2021) Since her thesis was to solve a classification problem, it is necessary to consider balancing dataset. However, in my research, I would not define it as classification because the recommendation system extracted relevant information simply based on production issue description. Even

though “REASON_CLASS” was selected as a target variable to compute performance result, it can be changed to other variables in terms of different evaluation interests. For example, if people are concerned about further detailed class than “REASON_CLASS”, P@K, AP@K, and MAP@K can be calculated based on “DEVIATION_REASON_CODE_ID” or “ROOTCAUSE_REASON_CODE_ID”.

The recommendation system with better performance relies on comprehensive models which are specifically effective on Finnish. Since the company owns businesses in multiple countries, the data in MES could be written in different languages. The models also work on other languages but require some minor modifications in code, such as changing package of stop words and finding suitable pretrained models to replace with.

The dataset was extracted from MES; hence this research is related to cold-start problem without online testing available. In reality, user reaction and expert feedback are valuable in terms of reliability and trustworthiness on business. A/B testing is the only way to measure its business value, but it takes more time and resources. At the current phase, it is impossible to implement online testing until the model can be built in MES. Once MES applies the feature of recommendation, the dataset will be expanding to a relatively good scale for model training in a certain period of data upgrading, so that the system can probably make suggestions more accurate and robust.

Another benefit of promoting recommendation online is to generate possibility of adopting collaborative filtering instead of content-based filtering, by engaging user information into MES. The system might require adding more data related to employees who create issue cases, such as their working sections, resolved issue cases and selected reference cases from content-based recommendation system, to discover the pattern or the relations between different users.

I have encountered some challenges on the journey of this research. The most vital one is language understanding, since I am dealing with Finnish that I am not good at. At the

beginning, I translated all the dataset into English via a pretrained model called “Helsinki-NLP/opus-mt-fi-en”. It took many hours and suspensions several times due to excessively long tokens in some documents. Then I switched the direction to concentrating on modelling for the original language of the dataset. Since Finnish is not as popular as English, there are limited packages and models for it. I had to apply stemming instead of lemmatization because Voikko installation in windows is problematic. Fortunately, deep learning pretrained models include Finnish lemmatization, ensuring the acceptable performance of recommendation system.

I am not able to use GPU model training in Anaconda Jupyter with personal computer, which is worthy to attempt with less time consumption. I would strongly recommend people who will oversee MES upgrading to train those models with GPU.

As a summary of discussions, the future works of the recommendation system could be taken place in the following aspects. Finnish lemmatization could be reached with Linux system and applied to test the performance of TF-IDF and Word2Vec models. The models could be optimized with more data available and by being tested online. The recommendation system should be durable to work on other languages with minor changes in scripts. It can also be transformed into collaborative filtering system instead of content-based, if data shifts into a new structure with more user information included.

6 References

A.M.Kuchling. (n.d.). *Regular Expression HOWTO*. Retrieved from python: https://docs.python.org/3/howto/regex.html

A.M.Turing. (1950, Oct). Computing machinery and intelligence. *Mind*, pp. 433-460.

A.Vashwani, e. a. (n.d.). Attention Is All You Need. *NeurIPS.*

Chandana, D. (2021, May 4). *Wayfair Recommendation System for furniture buyers*. Retrieved from Medium: https://medium.com/web-mining-is688-spring-2021/wayfair-recommendation-system-for-furniture-buyers-906c3f2d0427

Deutschman, Z. (2023, January 24). *Recommender systems: Machine learning metrics and business metrics*. Retrieved from Neptune.ai: https://neptune.ai/blog/recommender-systems-metrics

Hapke, H. (2019). Understanding, analyzing, and generating text with Python. In H. Hapke, *Natural Language Processing in Action.*

Ivens Portugal, P. A. (2018). The use of machine learning algorithms in recommender systems: A systematic review. *Elsevier*, 205-227.

Joshi, P. (2022, June 24). *Building a Recommendation System using Word2Vec: A Unique Tutorial with Case Study in Python*. Retrieved from Analytics Vidhya: https://www.analyticsvidhya.com/blog/2019/07/how-to-build-recommendation-system-word2vec-python/

Jung, A. (2021, 09 09). CS-EJ3311-Deep Learning with Python. Espoo, Finland.

Kangas, K. (2021). *Text analysis of handwritten production deviations.* Turku: Master of Science Thesis.

Kavlakoglu, E. (2020, 11 20). *NLP vs. NLU vs. NLG: the differences between three natural language processing concepts*. Retrieved from IBM: https://www.ibm.com/blogs/watson/2020/11/nlp-vs-nlu-vs-nlg-the-differences-between-three-natural-language-processing-concepts/

Krasnoshchok, Y. L. (2014). Extended content-boosted matrix factorization algorithm for recommender systems. *Procedia Computer Science*, 35:417-426.

LD, B. (2021, August 28). *Building a Text Recommendation System with Python*. Retrieved from Towards Data Science: https://towardsdatascience.com/build-a-text-recommendation-system-with-python-e8b95d9f251c

McFarland, A. (2022, June 25). *10 Best Python Libraries for Natural Language Processing*. Retrieved from Unite.AI: https://www.unite.ai/10-best-python-libraries-for-natural-language-processing/

Mikolov, T. C. (2013). Efficient estimation of word representations in vector space. arXiv preprint.

Nvidia. (2023). *Recommendation System*. Retrieved from Nvidia: https://www.nvidia.com/en-us/glossary/data-science/recommendation-system/#:~:text=A%20recommendation%20system%20is%20an,demographic%20information%2C%20and%20other%20factors.

Peng Xu, J. K. (2009). Using a Dependency Parser to Improve SMT for Subject-Object-Verb Languages. *Human Language Technologies.* Boulder, Colorado, USA.

PINECORE. (n.d.). Sentence Transformers:Meanings in Disguise. In *Natural Language Processing for Semantic Search.*

Rink, K. (2023, January 18). *Mean Average Preciosn at K (MAP@K) clearly explained*. Retrieved from Towards Data Science: https://towardsdatascience.com/mean-average-precision-at-k-map-k-clearly-explained-538d8e032d2

Robin van Meteren, M. v. (2000). *Using Content-Based Filtering for Recommendation.* Amsterdam: Computer Science.

Rocca, B. (2019, June 3). *Introduction to recommender systems*. Retrieved from Towards Data Science: https://towardsdatascience.com/introduction-to-recommender-systems-6c66cf15ada

Sanagapati, P. (2020). *Knowledge Graph & NLP Tutorial-(BERT, spaCy, NLTK).* Retrieved from Kaggle: https://www.kaggle.com/code/pavansanagapati/knowledge-graph-nlp-tutorial-bert-spacy-nltk

Scott, W. (2019, Feb 15). *TF-IDF from scratch in python on a real-world dataset*. Retrieved from Towards Data Science: https://towardsdatascience.com/tf-idf-for-

document-ranking-from-scratch-in-python-on-real-world-dataset-796d339a4089

sentence-transformers/paraphrase-multilingual-MiniLM-L6-v2. (n.d.). Retrieved from Hugging Face: https://huggingface.co/sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2

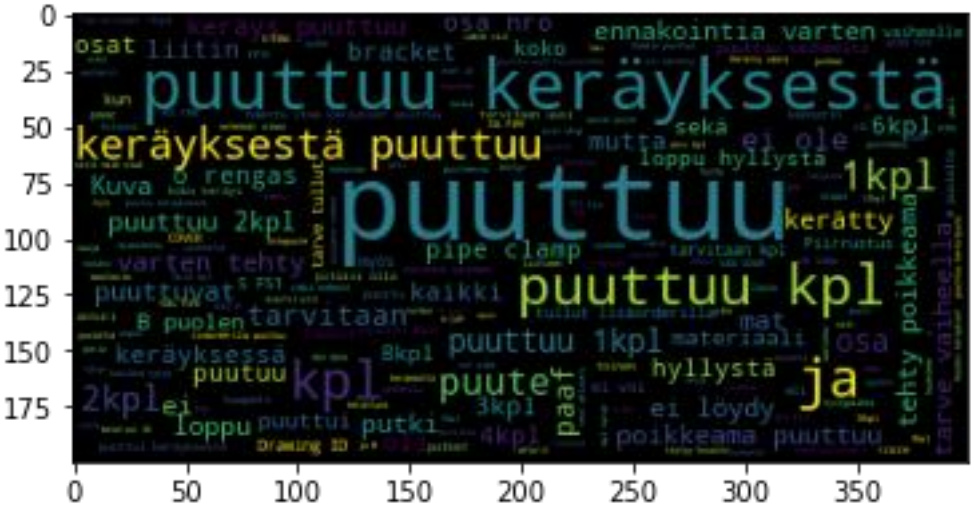
Wikipedia. (2023, March 17). *ChatGPT*. Retrieved from Wikipedia: https://en.wikipedia.org/wiki/ChatGPT

Wikipedia. (2023, March 9). *Natural language processing*. Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Natural_language_processing

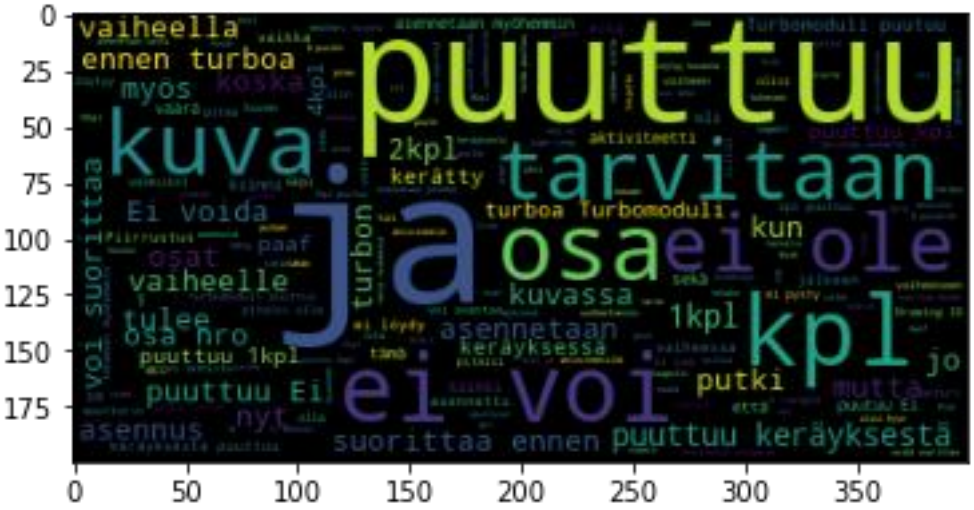
Appendices

Appendix 1. Wordcloud for "REASON_CLASS"

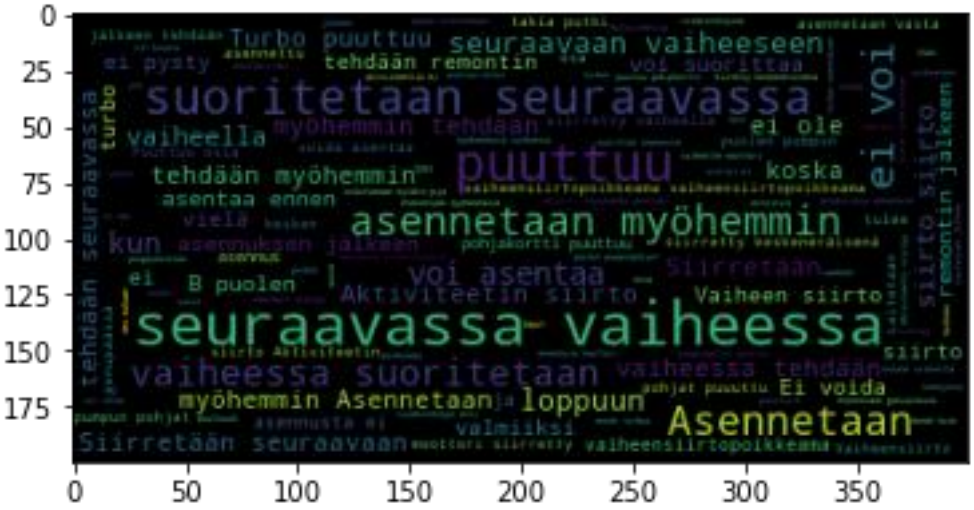
Reason class: Tuotteen laaduttomuus



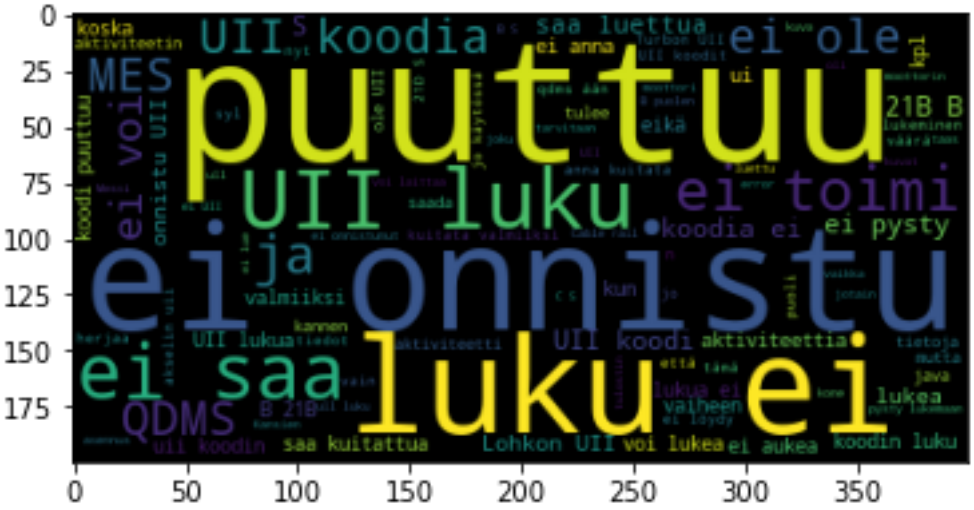
Reason class: Materiaalipuute



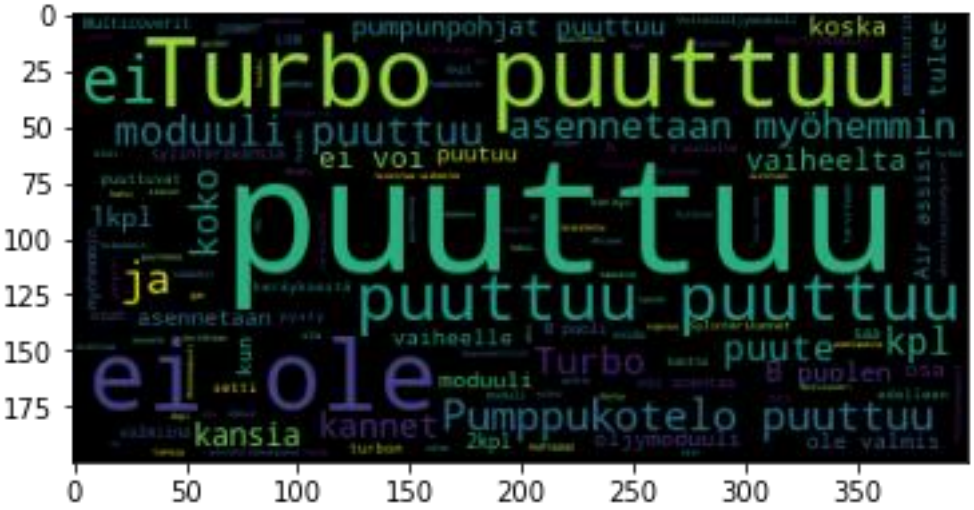
Reason class: Menetelmä



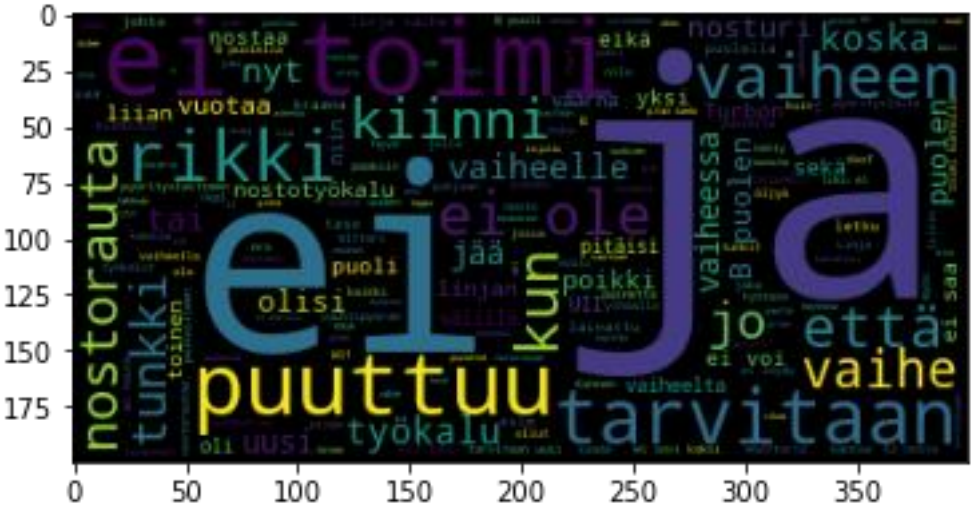
Reason class: Aktiviteetin siirto



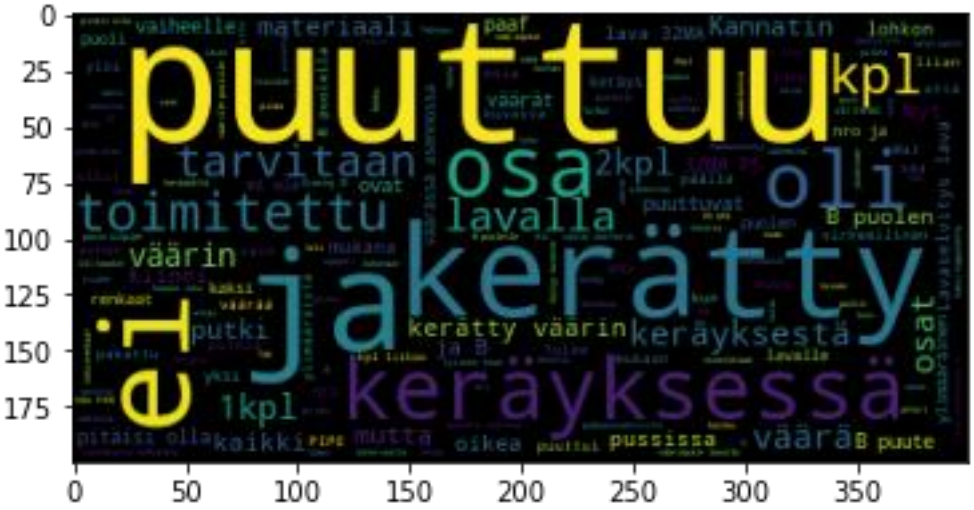
Reason class: ATK-järjestelmät



Reason class: Moduulipuute



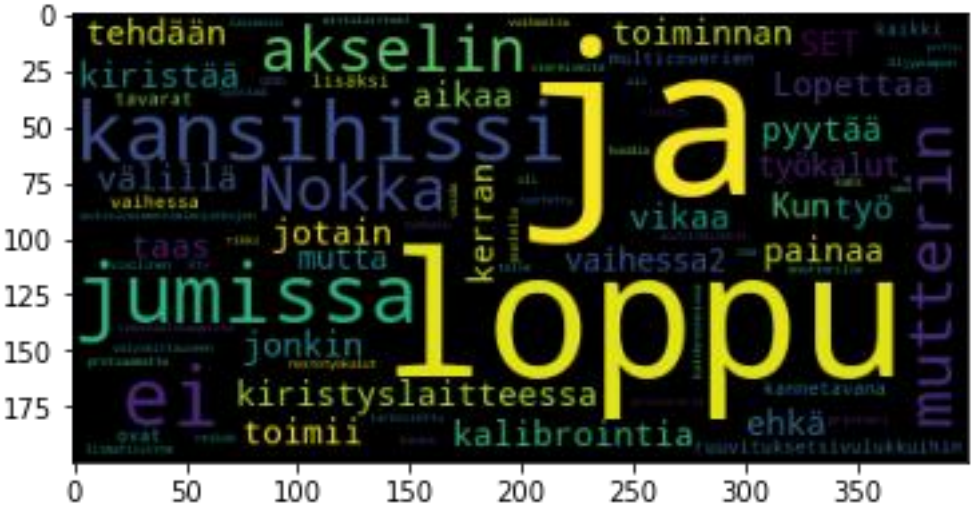
Reason class: Tekniikka: Koneet ja laitteet



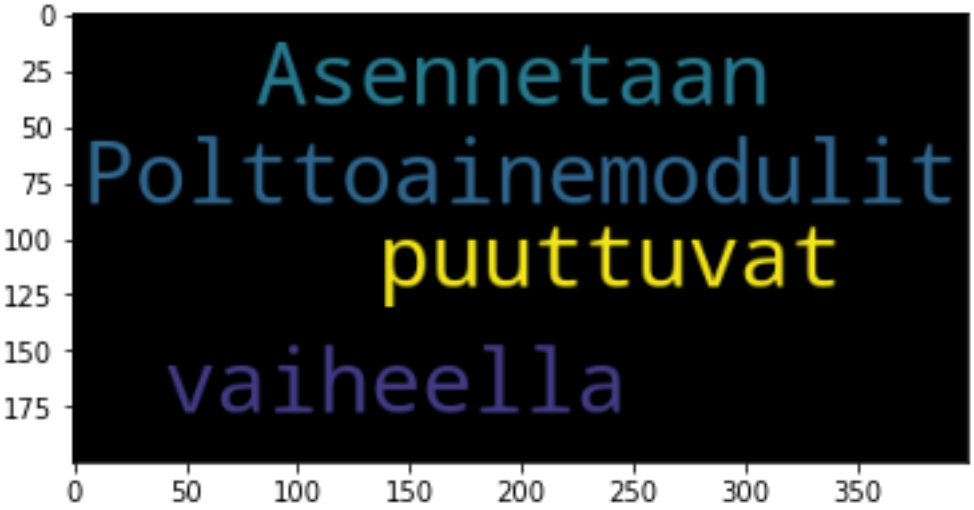
Reason class: Toimitusvirhe

Reason class: Puute

Reason class: MES



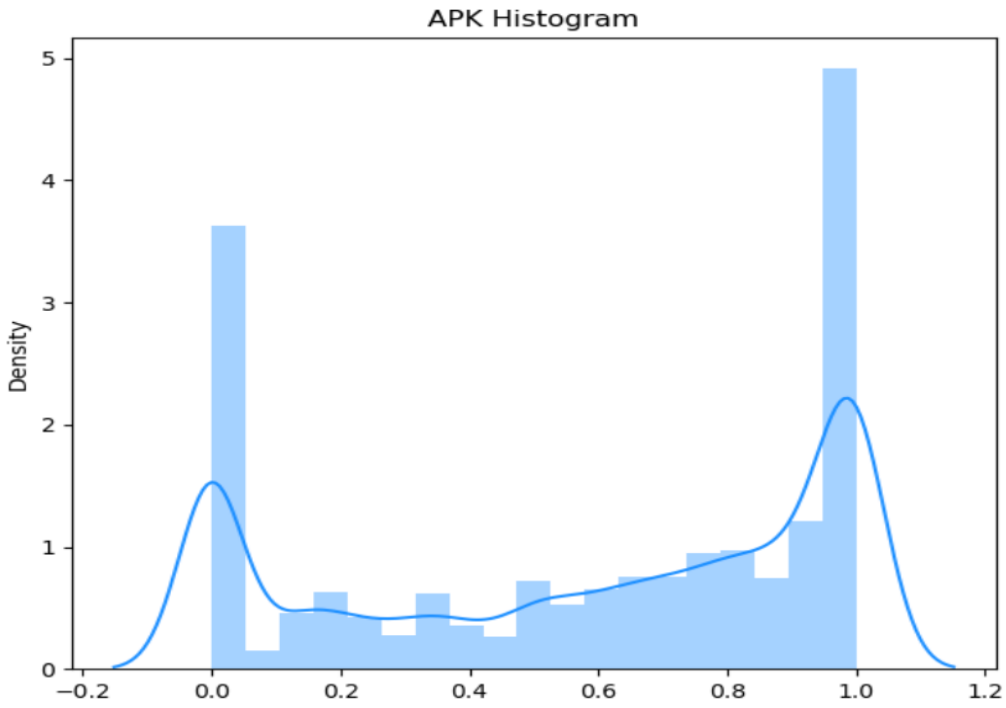
Reason class: Koneet ja laitteet



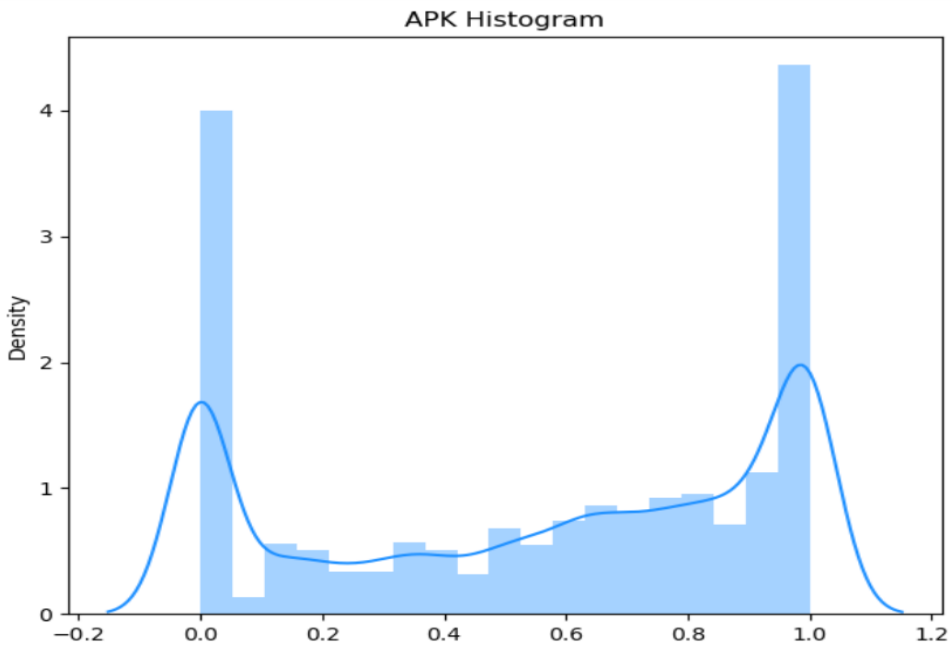
Reason class: Automatic Deviation For Missing Part

Appendix 2. Distribution of AP@10 for the models

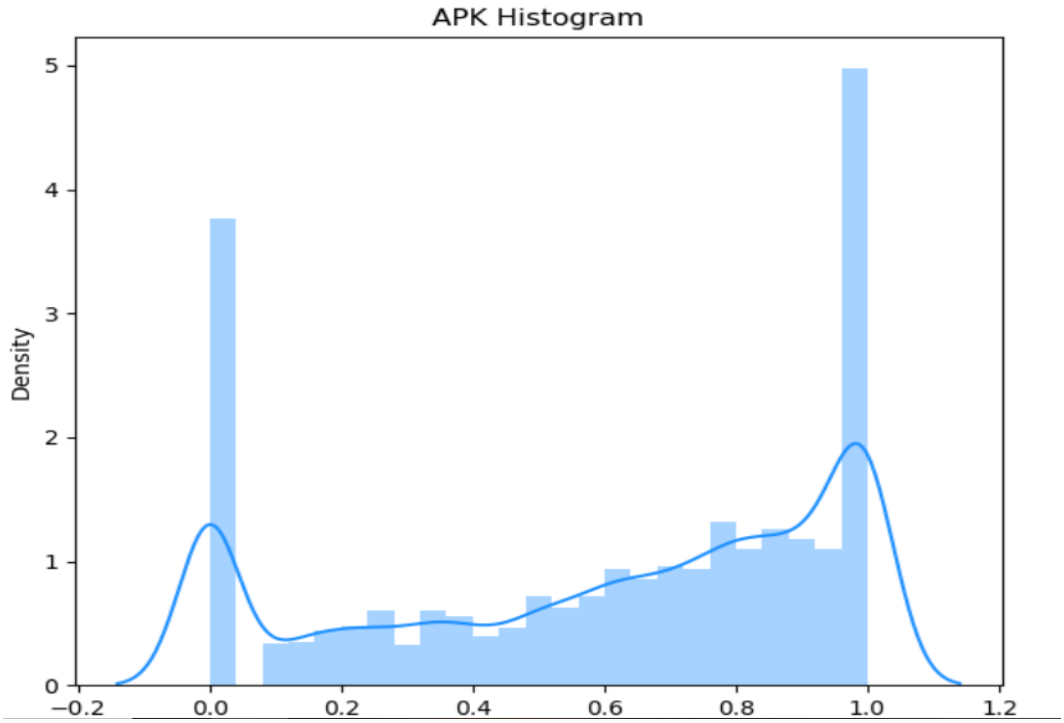
1. TF-IDF



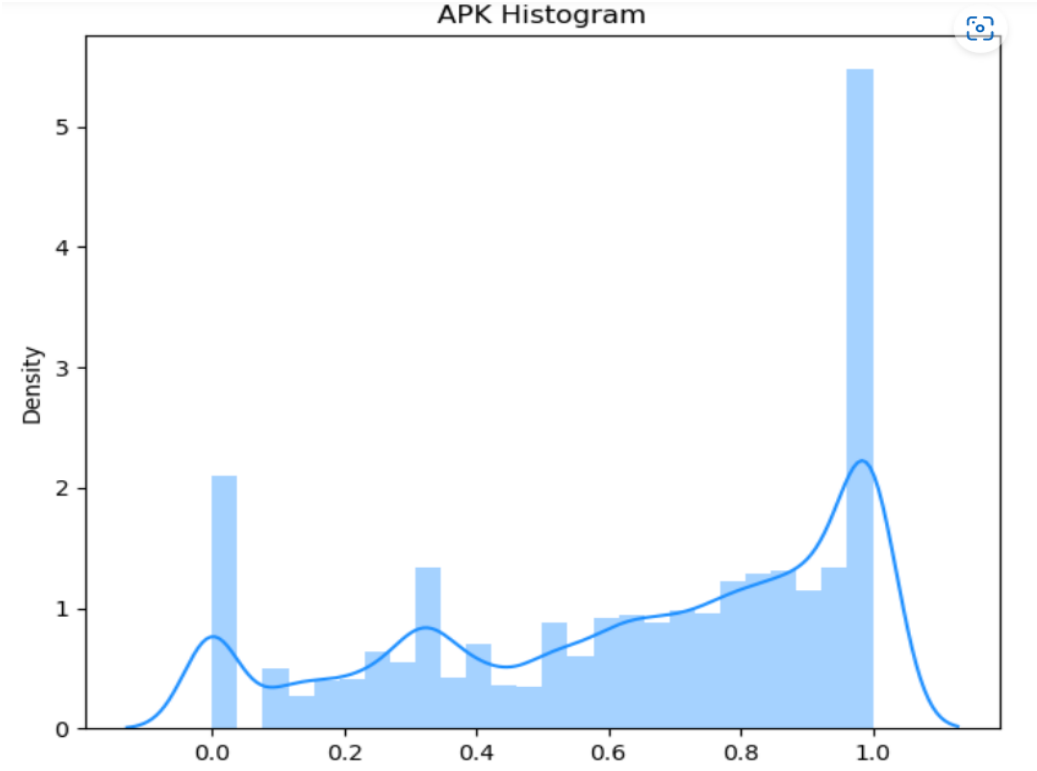
2. Word2Vec



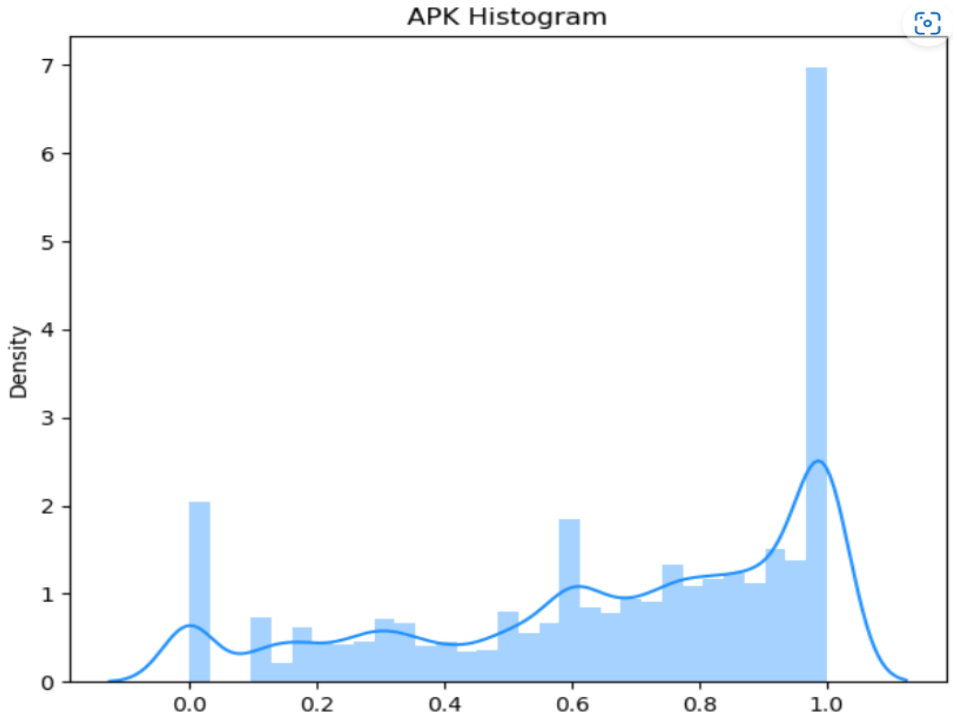
3. spaCy



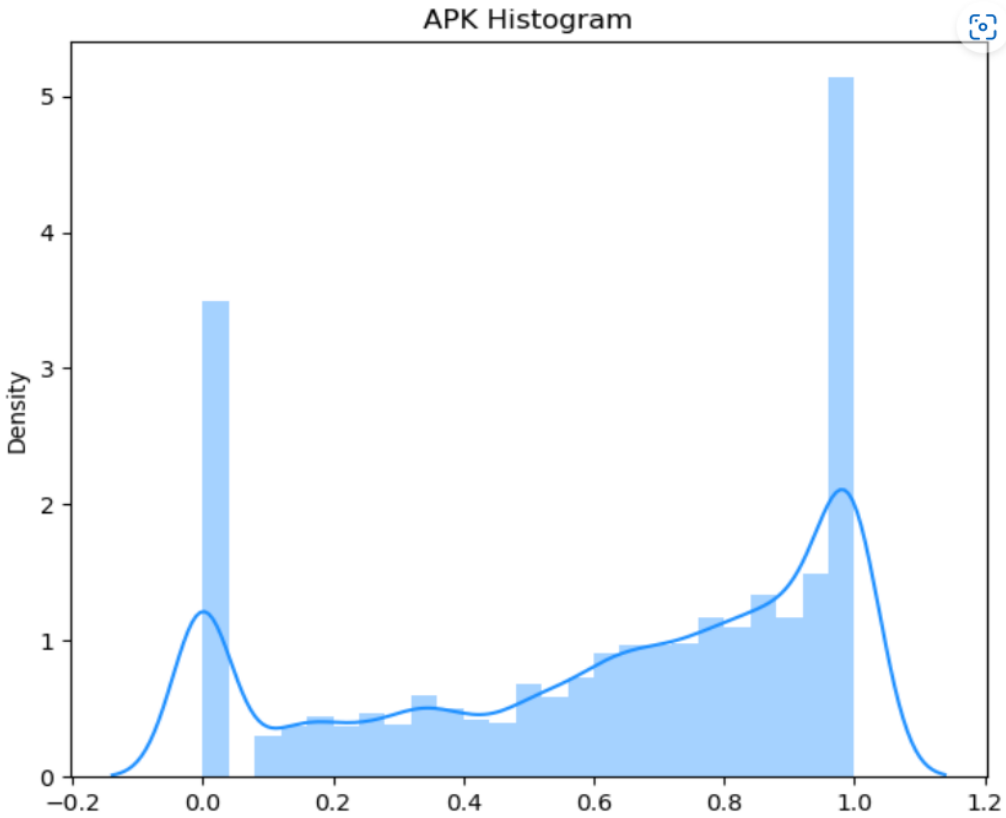
4. Multilingual Sentence Transformers



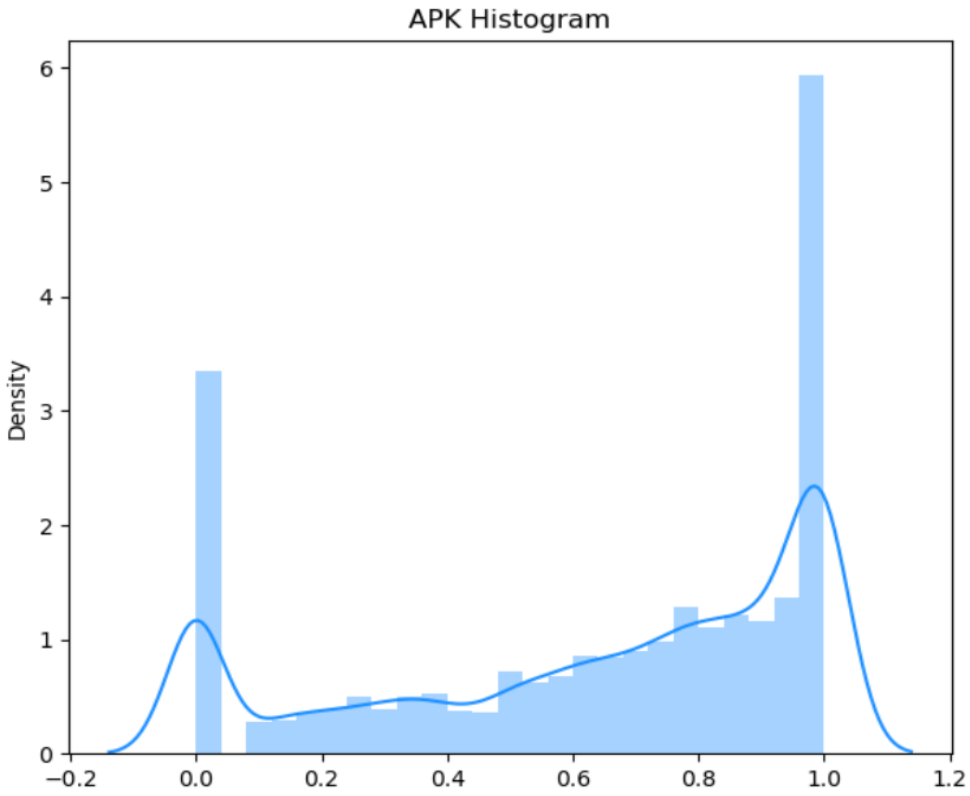
5. Turku-NLP-based Sentence Transformers



6. Multilingual SBERT

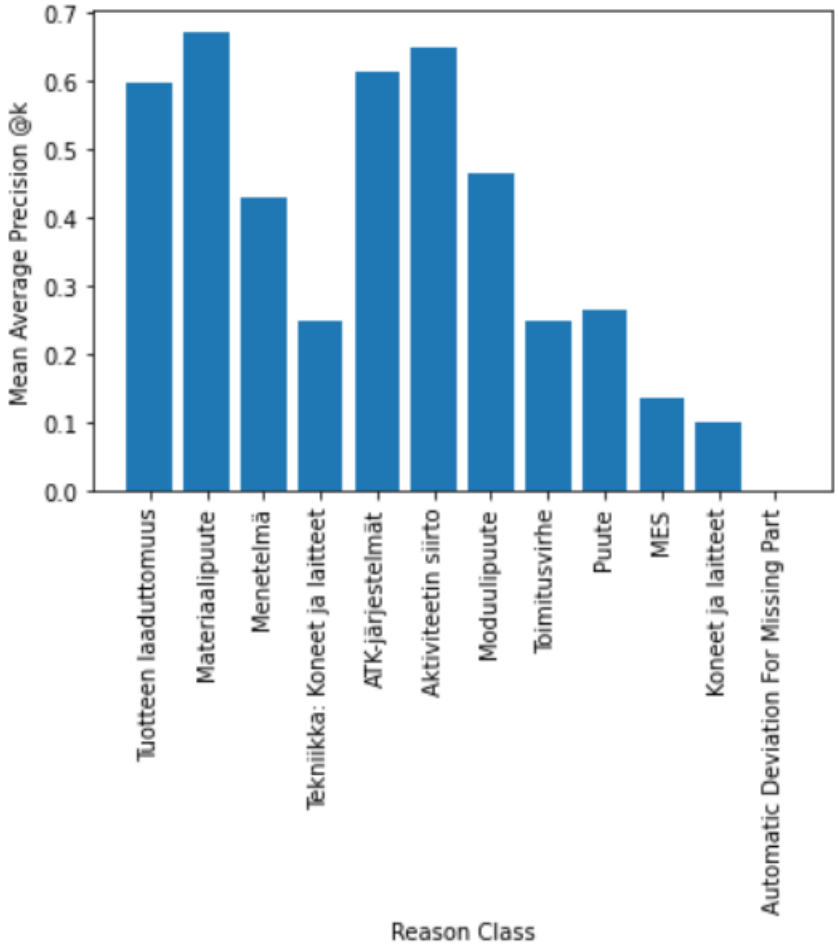


7. Turku-NLP-based SBERT

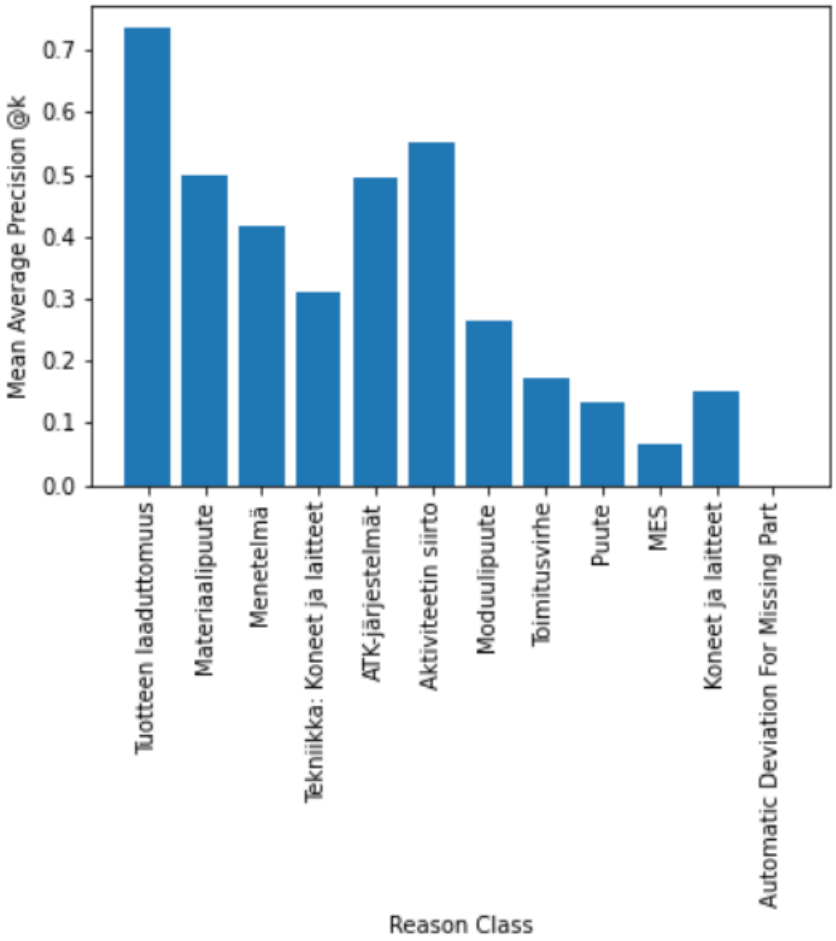


Appendix 3. Distribution of MAP@K based on “REASON_CLASS” (K=10)

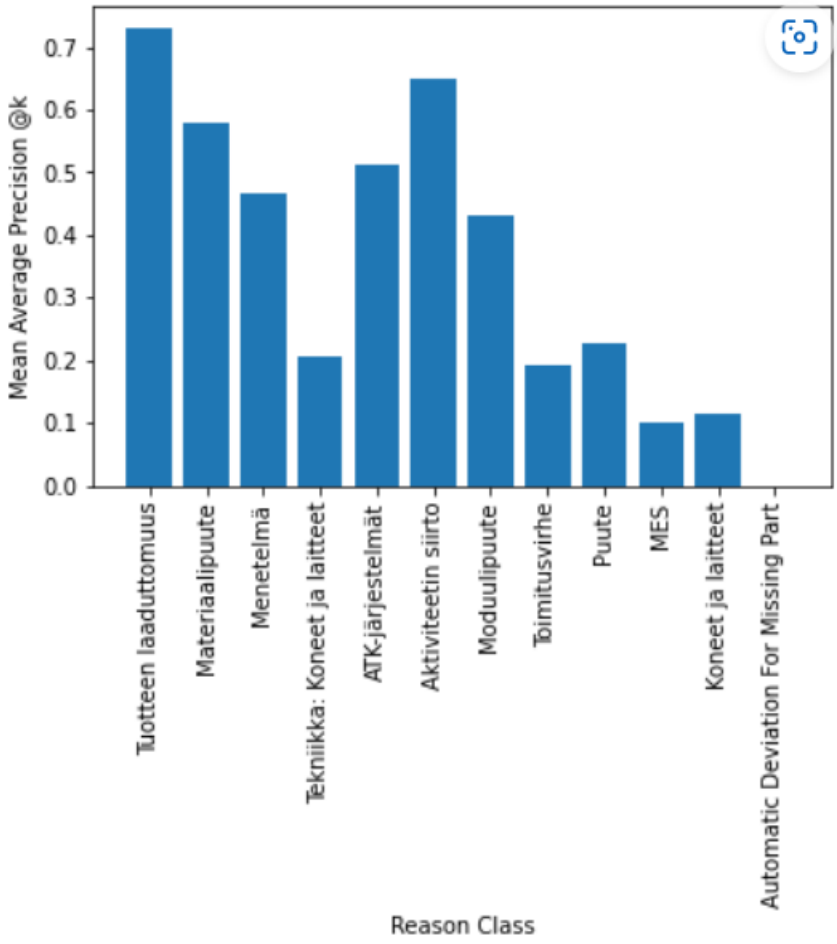
1. TF-IDF



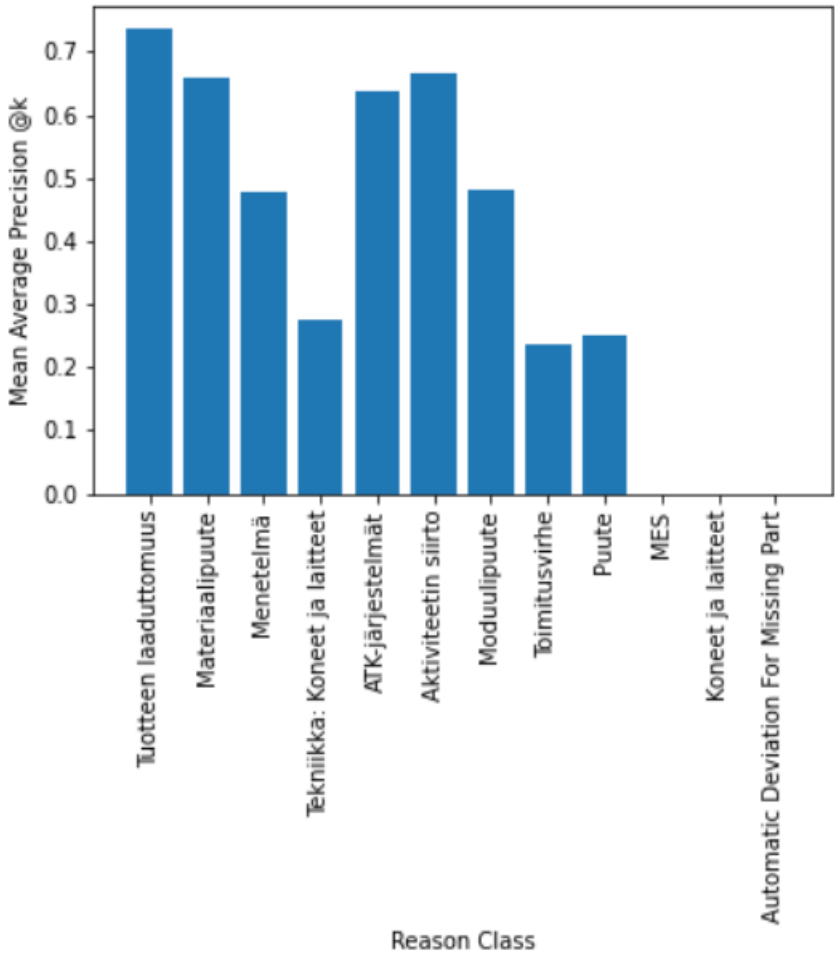
2. Word2Vec


Mean Average Precision @k
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
Tuotteen laaduttomuus
Materiaalipuute
Menetelmä
Tekniikka: Koneet ja laitteet
ATK-järjestelmät
Aktiviteetin siirto
Moduulipuute
Toimitusvirhe
Puute
MES
Koneet ja laitteet
Automatic Deviation For Missing Part
Reason Class

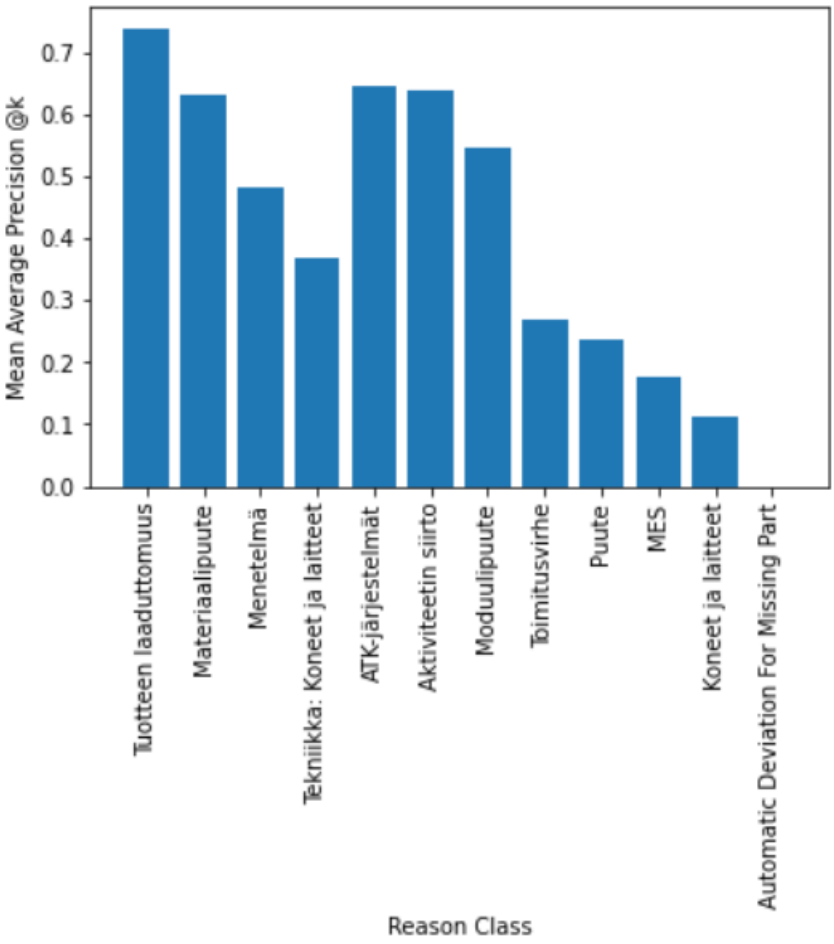
3. spaCy


Mean Average Precision @k
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
Tuotteen laaduttomuus
Materiaalipuute
Menetelmä
Tekniikka: Koneet ja laitteet
ATK-järjestelmät
Aktiviteetin siirto
Moduulipuute
Toimitusvirhe
Puute
MES
Koneet ja laitteet
Automatic Deviation For Missing Part
Reason Class

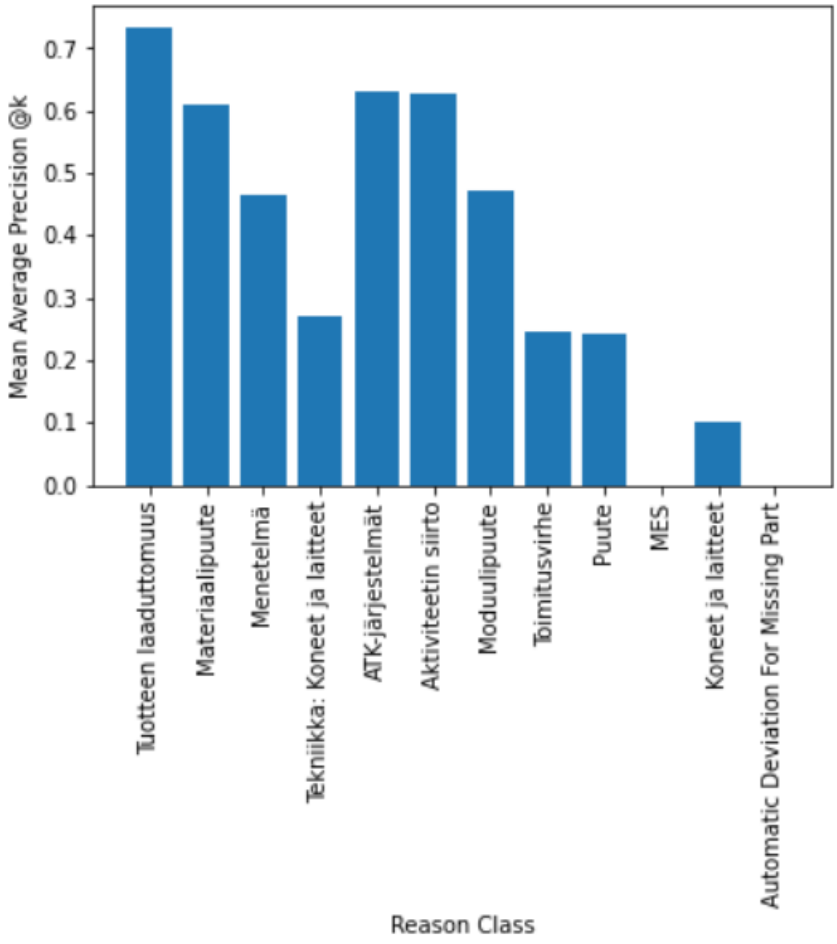
4. Multilingual Sentence Transformers



5. Turku NLP-based Sentence Transformers



6. Multilingual SBERT



7. Turku NLP-based SBERT