Master thesis

A QA-pair generation system for the incident tickets of SSC-ICT

University of Twente
Mick Lammers
m.r.lammers@student.utwente.nl
15th of March 2019

Supervisors:
Dr. A.B.J.M. Wijnhoven
Dr. F.A. Bukhsh

Company:
SSC-ICT, Dutch Ministry of Interior and Kingdom Relations

Abstract

The days of AI have begun, Artificial Intelligence becomes a common term in our vocabulary, even though most of us know and understand so little about it. It seems like only the huge and elusive companies like IBM and Google understand its use and potential fully.

In customer service, chatbots arise that answer customer questions based on most often manually crafted data structures called Question Answer-pairs, making companies look like one of the elite. However, what about those organizations that process so many questions that manual labeling is not an option? Should they remain old fashioned static servants that only react to their customer's inquiries that do not see a way to cater them proactively? The large companies provide the solution but with a price tag of millions of dollars. There must be something in between right? TopDesk, capping 80% market share in the Dutch incident management branch (Datanyze, 2019) does not see how.

In this study, we propose a low threshold QA-pair generation system using state-of-the-art technologies with the purpose of automatically identifying unique problems, and their solutions from a large and high variety incident ticket dataset of the nation-wide public IT Shared Service Center.

In order to achieve this, we researched the in related works applied components and techniques, and determined the for SSC-ICT best combination using identified characteristics of the dataset and organizational context. Furthermore, a set of component-based evaluation measures is designed in order to evaluate the different techniques and determine the best solutions. Then, a recommendation is provided with a system architecture, its use cases, and potential further improvements.

The result is a system consisting of 4 components: categorizational clustering, intent identification, action recommendation, and reinforcement learning. For categorizational clustering, we determine categorizational keywords using an existing Latent Semantic Indexing (LSI) algorithm to which we allocate the tickets using Levenshtein distance, which overcomes misspelling exclusions.

For the intent identification component, we compared two very different but state-of-the-art techniques: POS Patterns and Topic Modeling (LDA). After applying the evaluation measure, Topic modeling came out as the winner with a slightly lower QA-pair quality score, but higher improvement potential and a much higher ticket coverage rate.

The actions are cleaned, clustered and provided using a recommended application, a knowledge base application with reinforcement learning capabilities for use by the 40.000 customers of SSC-ICT. With enough feedback, the expected success rate of the system is about 50%. With further improvements, we believe this can lead up to 70-80%.

Other uses of the system's QA-pairs are Business Intelligence, FAQ extraction, and Anomaly Detection.

1 Introduction

IT Shared Service Centers are the beating heart of large organizations. They take on everything that has to do with the facilitation of IT: Personal computers, mobile devices, workplaces, servers, applications, VPN's. Now that more and more tasks and communication is done using computer devices, organizations are more dependent on them as well. IT Incident management, which manages the IT-related incidents within an organization and is a large part of Shared Service Centers' responsibility, is therefore crucial to the productivity of an organization.

As of now, incident management is performed in almost all service centers using a ticketing system. A ticketing system is a system in which incident calls or requests for service by users are registered by a service desk into a form which is called a ticket. The ticket is then either sent to the person within an organization that can act on the ticket or the person that knows the most about the context of a ticket. These ticketing systems do well what they are primarily meant for, and are especially very useful in large organizations in which alternatives for incident management like direct communication or e-mail would be inefficient.

However, what is often the case with these systems, is that the ticket data that they generate has excellent potential but often remains unused. The data often contains a description of the incident as well as the action that was performed upon this incident. This information could be used to create knowledge that could be used to automate service desk operator tasks or to be able to offer common solutions via a self-service portal or chatbot. In this research a system is designed by which the ticket data of a large Shared Service Center is used to create this knowledge in a manner that limits the amount of manual work as much as possible, using Natural Language Processing and Machine Learning.

The organization where the research is performed at is SSC-ICT. SSC-ICT is the IT Shared Service Center of 8 Dutch ministries. It supports about 40.000 civil servants that almost all have a company-laptop and phone as well as a virtual working environment. Furthermore, SSC-ICT provides service for over one thousand applications, and they have their own Data Center. All service-desks combined (phone(60%), e-mail (15%), physical(10%) and other (15%) generate around 30.000 tickets a month in ticket management system TopDesk.

Currently, SSC-ICT wants to increase its user satisfaction level. It is at a 6.7; their goal is a 7.0 at least. Monthly questionnaires show that this user satisfaction depends for a very high part on the customer service department, as well as on repeating complaints that are not taken care of. Management has spoken out and started a series of projects regarding being able to act more pro-actively instead of reactively on customer requests in order to increase the service satisfaction. One of these projects is meant to analyze the available data within the company with the purpose of finding use cases for it. This thesis research is part of this project.

When starting the project, in the first two weeks, we identified the data sources through interviews and calls. Very quickly, it was clear that the data of the service management system had the most potential to increase customer satisfaction and this data was yet unused. Literature research showed that application of Artificial Intelligence (AI) in the customer service management had great potential and was by far the number one researched subject in the field. However, this was more due to lack of research in the customer service field then due to the amount of research in AI, which is not that large.

The potential of implementing AI in customer support is very promising. According to recent research among 1082 senior IT-professionals from 11 European countries (ServiceNow & Devoteam, 2018), 72% of those that use AI in the customer service indicates to experience benefits from the technologies. However, less than a third of the customer service companies in the EU uses AI and only 22% of the Dutch customer service companies. Topdesk, the ticket management provider of SSC-ICT, has a whopping 88% market share but do not have any AI in their system, to show the differences.

Under AI in the customer service is understood virtual assistants and chatbot, Natural language processing tools, Sentiment analysis and text mining (ServiceNow & Devoteam, 2018). not have any AI in their system, to show the differences.

Furthermore, data analysis, as well as interviews with the managers of the service desk, has shown that 85% of all telephone calls to the service desk are first-line calls. They are thus answerable by the operator without him or her needing extra resources; this means that these tickets are rather easy to solve and therefore potentially automatically solvable or solvable by users themselves when provided with the right information. Thus, there are significant opportunities for automatization with AI at SSC-ICT.

A virtual agent can do all of the above and more. It would make the service be able to be available 24/7: also at night and the weekends. Furthermore, there would be no waiting times, and customers would receive consistent information, not having to rely on the operators' experience. Not to speak about the benefits of a business perspective like reduction in service operator's cost.

However, complete AI systems like IBM Watson or Amelia of IPSoft are expensive. Estimates point towards investments of multiple millions of dollars for a company like SSC-ICT. Also, they require substantial changes in infrastructure, as they built on learning from feedback, namely reinforcement learning. Training such a system from scratch takes at least 12 months to catch up with the organization's processes and be more efficient than without such a system. A leap this far, costly and with little transparency is something that not many organizations are willing to take.

However, we think that this is not where it ends. There is an area between a fully automatic cognitive AI system and a static ticketing system. What is needed is a first step on the ladder towards AI, a low threshold system that shows quick benefits of applying AI in customer service and is transparent in its results. SSC-ICT has the perfect environment to build this, due to its scale, quick win potential and number of users. This research describes a low threshold bootstrapping system (Dhoolia, Chugh, Costa, Gantayat, et al., 2017) for AI in customer service that serves as a foundation for continuous improvement.

1.1 Problem statement

How can AI make use of ticket data? The tickets of SSC-ICT consists among other fields on the description of the problem and the action that is performed on the problem by the service desk operator. What AI techniques can do is identifying unique problems from the tickets, compare them to similar problems, and provide suitable action, based on history, all the while without much manual effort. There are different components in this process needed due to the distinction between problem and solutions and the matching between those. A component that large cognitive systems like IBM Watson are very advanced in is reinforcement learning. Reinforcement learning is learning from feedback mechanisms, and it requires much feedback. In this research, we focus on "bootstrapping" the cognitive system by identifying problems and matching solutions, i.e., generating Question-Answer pairs (QA-pairs). The reinforcement learning part is given a start with but is not developed in-depth due to the need for long-term feedback and continuous improvement. In the next chapter, we formulate the research scope in a research question and sub-questions.

1.2 Research questions

What is a “State of the Art” QA-pair generation system for incident management of SSC-ICT?

1. **What components, techniques, and characteristics of QA pair generation systems are used in related works?**

A literature review will be performed to identify all available components and techniques in QA-pair generation. We perform a literature review on a wide array of AI applications for ticket management systems and extract the general topics which we will describe in chapter 2. Next, from this same literature review, we extract a short-list of the most similar research cases to this research, and we will analyze them thoroughly. We provide summaries of these related works in chapter 2.5, and we accumulate requirements from them for our system in chapter 2.6.

2. **What potentially useful, other techniques are there?**

Apart from literature, online communities are, especially in Data Science, a great way of collecting inspiration. In chapter 2.7, we accumulate all the techniques that we use in this research, and we will explain how and why.

3. **What are the characteristics of the SSC-ICT dataset?**

Based on this research question we analyze the dataset of SSC-ICT, with the perspective of building the system. We analyze the data fields, their use, we describe the ticket input process, and how the final dataset is composed.

4. **How can QA pair quality best be measured?**

To evaluate the system and to be able to compare the results of different techniques, measures for the quality of the QA-pairs are needed. In the literature review among related works, the encountered evaluation techniques are evaluated. Furthermore, we apply literature research on evaluation techniques specific to the components of the system.

5. **What is the minimal quality level needed for the evaluation corpus to produce relevant performance measures?**

Setting a minimum quality level helps to see the system’s results in perspective. We base the quality level on achieved results of related works as well as on prognoses of field experts.

6. **How can QA pairs best be used at SSC-ICT?**

QA-pairs have multiple use cases. Based on the characteristics of SSC-ICT we recommend one or two use cases. Furthermore, we will provide a prototype version of such an application, based on the ticket data of SSC-ICT.

1.3 Research approach

For this research, we chose to use a custom research framework. Our framework is based on the Cross Industry Standard Process for Data Mining (Crisp-DM). This model is a widely used methodology for data mining projects and has use cases in projects within immature research fields. Furthermore, this model is very practically oriented rather than theoretical which suits this research project well.

In figure 1 the dimensions of the Crisp-DM model are provided along with their generic tasks, this helps to understand the dimensions better. In figure 2 the adapted version of the Crisp-DM model is visualized. In this version, we combine data preparation and data modeling due to the synergy of these tasks in Natural Language Processing (NLP). Furthermore, we added another dimension, namely determining the high-level architecture. We did this because NLP systems other than most data mining projects, often consist of a pipeline of components, rather than one, that have different input and produce different results.

In the next paragraph, each of the dimensions is described in more detail as well as where in the report the elaboration on it is described.

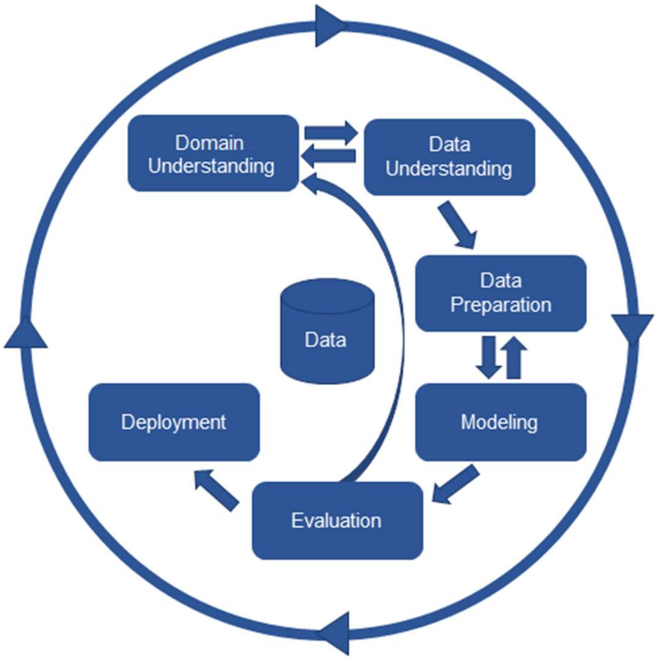


Figure 1: Generic tasks of Crisp-DM Reference model (Chapman et al., 2000)

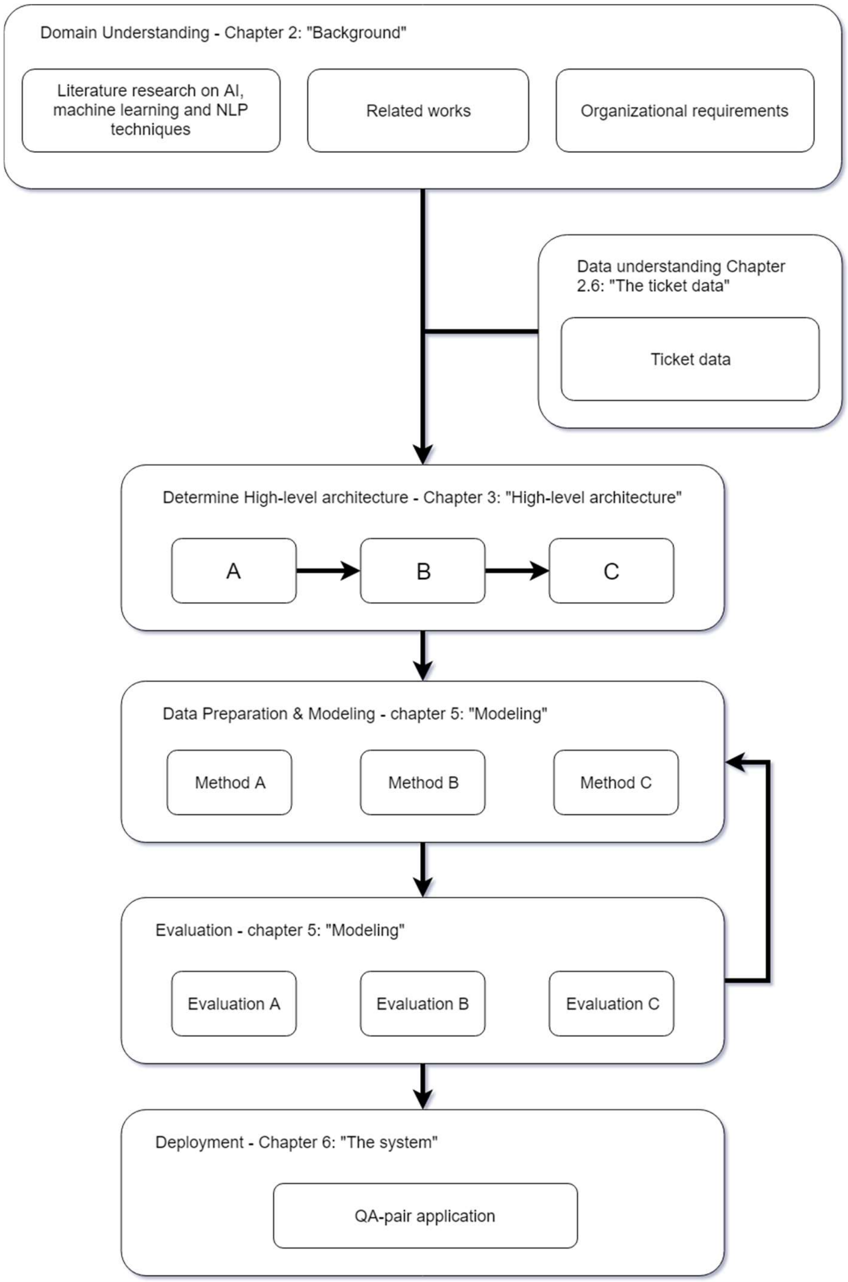

Domain Understanding - Chapter 2: "Background"
Literature research on AI, machine learning and NLP techniques
Related works
Organizational requirements
Data understanding Chapter 2.6: "The ticket data"
Ticket data
Determine High-level architecture - Chapter 3: "High-level architecture"
A
B
C
Data Preparation & Modeling - chapter 5: "Modeling"
Method A
Method B
Method C
Evaluation - chapter 5: "Modeling"
Evaluation A
Evaluation B
Evaluation C
Deployment - Chapter 6: "The system"
QA-pair application

Figure 2: Adapted version of CRISP-DM research approach

Domain understanding

In this first phase, the research domain is explored and understood. A literature review is applied to find similar cases, to scope down the research domain as well as to find technologies and components that are potential candidates for this research project. We describe similar cases, components, and technologies in chapter 2: Background.

Data understanding

Data understanding is about understanding the potential and limitations of the data regarding its expected results. We describe this topic in chapter 2.6: The ticket data.

Determine high-level architecture

This dimension is about determining what components are best for the system. Once this is determined, it remains as is and the modeling of processes and evaluation can advance. In short, it is the foundation of the system. This dimension is described in chapter 3: High-level architecture.

Data Preparation & Modeling

Modeling is for this research the process of choosing, designing, building and evaluating of models and algorithms with the goal of reaching the expected results. This process, as well as visualized architectures, are described in chapter 5: Modeling.

Evaluation

Evaluation criteria are defined componentwise. For each of the design iterations, we measure and evaluate the effectiveness of the solution using the criteria. In chapter 4 the criteria are defined, and in chapter 5 they are applied.

Deployment

In chapter 6, the final system is described and the performance is determined and compared to the minimal quality level, which is described in chapter 6 as well.

1.4 Research taxonomy

- **Intent**: an intent is an identified problem or the Question in Question & Answer pair.
- **Short description:** a field of the ticket dataset containing a summary of the problem, used for identifying the intent
- **Categorical clustering:** clustering on the highest level
- **SSC-ICT**: Shared Service Centre – ICT
- **AI**: Artificial Intelligence
- **NLP**: Natural Language Processing, an AI subject for natural language
- **Deep Learning**: Machine Learning using neural networks
- **QA-pair:** A question-answer pair, a combination of a question and a suitable answer.
- **Customer/user:** The Dutch civil servants
- **Operators:** Service-desk employees

2 Background

This chapter describes background information regarding this research. First, we describe the main domains of Artificial Intelligence in customer service. Then, we describe the evolution of AI systems based on a literature review among 50 articles regarding AI systems in the customer service. Followed up by common applications of QA-pairs which is based on the literature review. After that, common techniques in QA-pair systems are summed up. Next, we describe the ticket dataset of SSC-ICT. Then, we describe related systems to this research system. We summarize these articles and extract characteristics from them. These characteristics are then applied to SSC-ICT.

2.1 Artificial intelligence, Machine Learning, and Natural Language Processing

Russell & Norvig (2013) define Artificial Intelligence in four different approaches: machines that act humanly, machines that think humanly, machines that act rationally and machines that think rationally. For this research we will use the definition of machines that act rationally, or “Computational Intelligence is the study of the design of intelligent agents”. This definition is most fitting because in this research an agent is designed that acts rationally; it offers rational solutions to problems.

Natural language processing (NLP) is a big part of AI that is used in the customer service. NLP is defined as all techniques used for the processing of natural language text. Since all explicit knowledge is stored in either digits or natural language, natural language processing is a big subject within AI.

Natural Language Processing consists of but is not limited to reading, extracting information, creating new information and generating natural language. NLP makes use of techniques that are part of Machine Learning, which is the other big subject within Artificial Intelligence. Machine Learning can be another topic, Deep learning, which can be seen as a subtopic within Machine Learning is also often used in combination with NLP.

Summarized, figure 3 in which the subjects within AI, Natural Language Processing and Machine Learning and their overlap are visualized, explains the definition of these topics best for this research.

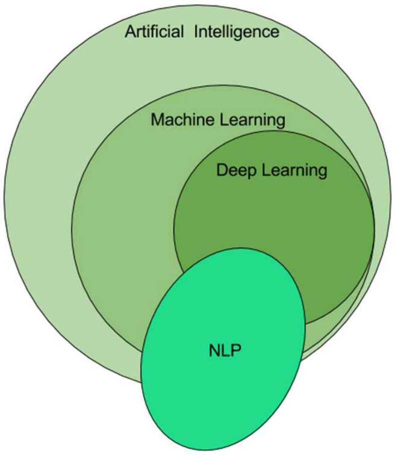


Figure 3: AI, ML, DL and NLP

2.2 QA-pairs

The results of the system that we describe in this report are what are called Question Answer (QA) pairs. QA pairs are a combination of a question and an answer. In incident management the question is often referred to as “intent”, we use this term in the rest of this report as well. The intent is the user’s intent for creating the ticket. The answers are called actions, resolutions or just answers, in this report we will use the term “action”, because this term is also used in the TopDesk ticketdata. The combination of the intent and the action we call the QA pair. The idea behind the creation of QA pairs from ticket data is that the tickets with the same intents are clustered and the applied actions on the tickets are provided as potential answers.

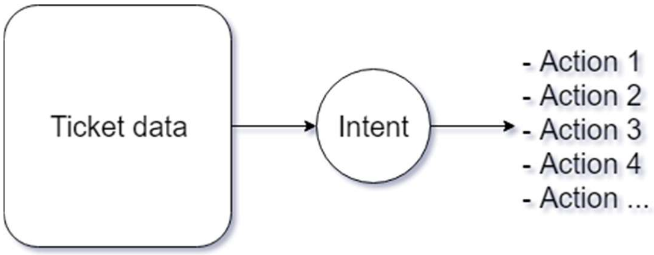


Figure 4: Intents and actions as QA-pairs from ticketdata

2.3 Applications of QA-pairs

In this paragraph, the different applications of AI in the customer support service are discussed. This list is built based on a literature review that we performed among 50 articles regarding AI applications in Customer service. The list of materials can be found in Appendix A. The literature research methodology is found in Appendix B. The list is the following:

- Chatbot/virtual agent
- Knowledge base
- Business Intelligence
- Anomaly detection

A chatbot or virtual agent is a system that can answer questions of users and drill down with a specific follow-up question in a chat environment. A knowledge base is an internally used system in which complex low-level information is stored that can be called intuitively.

A Business Intelligence system is a decision-making system used by management or analysts to get a high-level perspective on a particular aspect of an organizations practice.

Anomaly detection is a technology in which major incidents are automatically detected based on triggering of certain thresholds that are set based on AI generated features.

2.4 Techniques in QA-pair generation systems

In this chapter, we summarize and explain the techniques used in scientific research for AI systems in customer service. This chapter serves to provide a global view of the topic. We describe the techniques that are prevalent in pre-processing of text. Furthermore, we describe techniques that are common for clustering text.

Why pre-processing, clustering and synonyms? Pre-processing is important for getting the data in the right form. Clustering is essential for classifying. Synonyms are important for normalization of text so that clustering can be applied more successfully. In this chapter, we describe these techniques that are used further in the report. It provides an overview of the subjects.

2.4.1 Natural Language Pre-processing

Natural Language pre-processing is the process of preparing and normalizing text for machine learning processes. The following are the most common pre-processing techniques: tokenization, capitalization, stop-word removal, stemming, lemmatization, spelling correction, noise removal, n-gram creation, word embeddings, and part-of-speech tagging.

Tokenization is the process to split sentences into words, of which the collection is commonly called a “bag of words”. To be able to compare all of these words, they are turned into lowercase words. Next, stop words can be removed for topic extraction, as stopwords are not contributing to this end and are consequently considered as noise. Stemming is a process in which the last characters of words are cut off using a simple algorithm removing common prefixes. This process further increases the normalization of words. Next to stemming there is also a more advanced variant called lemmatization. This process is mostly based on deep learning and brings back words to their root form. For instance: is > be, and bought > buy. Spelling correction is mostly performed using an edit-distance or Levenshtein algorithm. This algorithm computes the number of operations to change one word into another. Then noise removal is typically the process of removal of specific system or text-type related characters like timestamp or mail-signatures. Noise removal can be performed using many different techniques ranging from regular expressions to deep learning. N-gram extraction is the process of finding common sequences of n-amount of words. It is used to find topics within sentences or to find common concatenations of words. It can range from frequency-based calculations to advanced deep learning models. Finally, word embedding is the most abstract technique in this list as it is the transformation of words into digits with the purpose of preparing text for Machine Learning. The most common word embedding technique is used in more than 80% of search-related systems is TF-IDF (Term Frequency-Inverse Document Frequency). TF-IDF is a vector for a word depicting how often the word appears in a document to how often it appears in a larger set of documents. Thus the less often the term occurs in other documents, the higher its TF-IDF score.

Finally, Part of Speech (POS) tagging is a Natural Language Process of labeling words with their grammatical word-form. POS tagging is either done based on a library or on an algorithm that uses syntax and positioning and uses Deep Learning or a combination of both.

There are numerous applications of POS tagging. The identification of word forms can help for instance with finding entities or operations as most entities are nouns and most operations are verbs. Entities and verbs can, in turn, be used to summarize sentences.

2.4.2 Clustering

Clustering is a grouping name for all technologies that group data according to similar characteristics. In Natural Language Processing, the input is often word embeddings which are explained in the previous paragraph “Pre-processing”.

The most established text clustering methodology that uses word embeddings is Latent semantic analysis (LSA). In LSA, a term-document matrix is constructed using the word vectors for all the terms and then using a methodology called Singular Value Decomposition patterns and relationships among these terms are identified, and concepts can be compared.

One other common and recent use of TF-IDF for clustering documents is topic modeling, or Latent Dirichlet Allocation (LDA). LDA is an unsupervised algorithm that essentially determines a set of topics over a corpus and provides a weight of accordance of each document to each topic. This way it can identify dominant topics.

Now these word embedding clustering methodologies are in essence all statistical. There are however also syntactical clustering methodologies. For these methodologies, no data is needed as they appoint a label to data based on only that data itself. The most common syntactical clustering methodology is that of POS patterns in which patterns of specific Part of Speech are recognized as containing important aspects of a sentence.

2.4.3 Synonyms

Synonyms are an important challenge in customer service AI systems.

In synonym detection, there is a separation between domain-specific and general synonyms. General synonyms are synonyms of ordinary daily used words, domain-specific synonyms are only found In their respective domain, examples are names of applications or processes.

General synonyms can be identified using large lexical databases which are almost always open-source. Domain synonym detection is not possible using lexical databases, as the keywords are generally domain-unique and therefore not found in lexical databases. For this task, there are no tools available as of yet as well. However, many research has been done on this topic; different technologies are used with different results on different types of text. For one, word2vec is a technology created by Google in 2013. This technique makes use of word vectors and two-layer neural networks that compute similarity based on linguistic contexts of words. Its advantage is that it is rapid and that the technology is readily applicable. However for it to be accurate, large amounts of text (more than 10 million words) are needed, preferably with documents with multiple sentences.

Another technique that applies to domain synonym detection is from S. Agarwal et al. (2017). They designed an entity similarity algorithm that computed similarity based on similar operations among entities, it would be especially useful for short text documents and needs a medium-sized corpus. Its’ downside is its speed and inaccuracy for documents with multiple sentences. It was designed because other techniques, like word2vec, created too much noise on their dataset.

2.4.4 Reinforcement learning

Reinforcement learning is the third dimension of machine learning next to supervised learning and unsupervised learning. It is a very general problem description for the goal-oriented interaction of an agent (system) with the environment (user) as is shown in figure 4. The agent provides the best form of action it knows to a situation in the environment, and the environment sends back a response which is interpreted by the agent as either positive or negative feedback from which it can adjust its future action regarding similar situations. We call it general because there are so many ways by which reinforcement learning can be applied, the most common one being dynamic programming, and recent research is diving into using deep learning for reinforcement learning in NLP (Sharma & Kaushik, 2017).

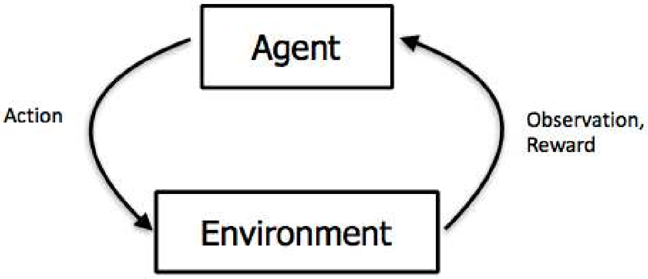


Figure 5: Reinforcement learning

2.5 Related works

In this chapter we summarize and analyee the most related works to this research case from a literature review among 50 articles regarding AI systems in the customer service. All the systems that we descreibe are QA-pair systems from incident tickets. We have not found other relevant systems in the scientific literature.

The articles are discussed below and are in order of relevance to this research.

(1) In P. Dhoolia et al. (2017) a cognitive support system is designed for a specific client that has 450 factories and operates in 190 countries. The system is aimed to answer level-1 and level-2 support questions associated with IT applications used by enterprise WW users. In order for that system to work effectively, they attempt to extract question and answer pairs from tickets with the goal of bootstrapping a cognitive system. For extracting the intents, they used a combination of n-gram and Lingo techniques (Osinski, Stefanowski, & Weiss, 2004), as well as field experts to manually identify intents. These intents were then used to match live tickets to.

To identify intents from live tickets they applied the following processes: 1) group the repeating or similar tickets into problem clusters, 2) select the appropriate cluster, and 3) extract the representative question-answer pair from the cluster. They did this by parsing user questions to extract business entities and actions into a knowledge graph. During a conversation with the user, they explore the neighborhood of the sub-graph in order to find probable intents.

Furthermore, continuous feedback learning was applied to continuously improve the system. When helped, the customer could leave feedback regarding the process which information was placed in a human expert verification queue and applied after approval by the human expert. They made use of the feedback in 6 different ways: identifying question variations, identifying probable new questions, identifying flaws in the intent disambiguation process, learning new intents, learning the new mapping between knowledge units and intents

In the end, 130 support intents were identified in the domain. The system was able to answer 50% of the questions.

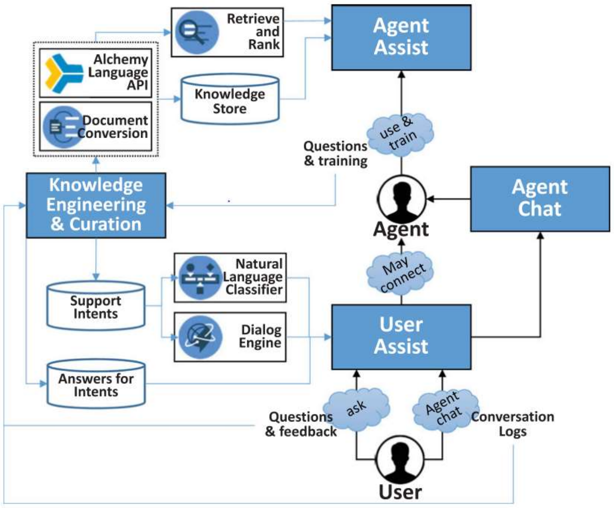


Figure 6: System architecture (Dhoolia, Chugh, Costa, Gantayat, et al., 2017)

(2) In S. Agarwal et al. (2017) a cognitive system was designed by researchers from IBM for the use in service providers' service desks. The knowledge extraction processes applied is divided into three steps: problem diagnosis, root cause analysis, and resolution recommendation. For the problem diagnosis process, logical structures in ticket texts were identified to pre-classify tickets into either simple or complex groups. Next, a classification engine based on a support vector machine with a Radial Basis Function is built. To train this engine, 5000 problem tickets were manually labeled by experts into 15 categories.

For the Automated Root Diagnosis (RCA) process linkages between a problem and its probable cause were extracted. These linkages are based on using features such as time of occurrence and similarity of the IT entity on which they occurred, as well as common terms in the text descriptions of the problem and change (S Agarwal et al., 2017).

For the resolution recommendation, three processes were used: identifying the action phrases from the resolution texts, deducing domain dictionary and semantic similarity and finally building the summary phrases. Identifying the action phrases is needed to focus on the right information in large texts. This was done by determining the most relevant POS patterns and finding phrases that match these patterns. For deducing a domain dictionary, a custom algorithm was built that identified similarity based on common operators on entities and the other way around. Action phrases were then built by combining entities and operations in a summary phrase.

The system was able to find a solution to 67% of incoming tickets. The system was able to reduce the time needed to solve a ticket by half by being able to offer probable solutions from 70 minutes to 35 minutes. Dataset was 1000 IT tickets.

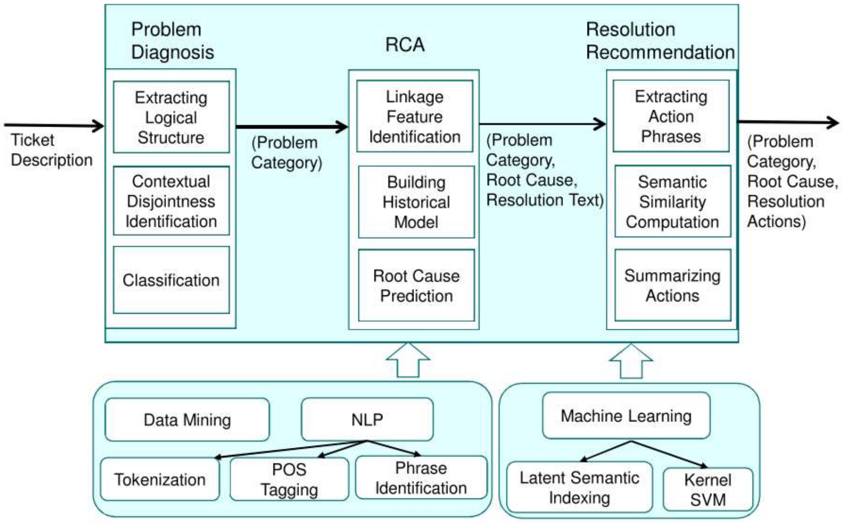

Problem Diagnosis
RCA
Resolution Recommendation
Ticket Description
Extracting Logical Structure
Contextual Disjointness Identification
Classification
(Problem Category)
Linkage Feature Identification
Building Historical Model
Root Cause Prediction
(Problem Category, Root Cause, Resolution Text)
Extracting Action Phrases
Semantic Similarity Computation
Summarizing Actions
(Problem Category, Root Cause, Resolution Actions)
Data Mining
NLP
Tokenization
POS Tagging
Phrase Identification
Machine Learning
Latent Semantic Indexing
Kernel SVM

Figure 7: System architecture (S Agarwal et al., 2017)

(3) In Mani et al. (2014) an approach is proposed to automatically analyze problem tickets to discover groups of problems being reported in them and to provide meaningful labels to help interpret these groups. The method is based on incorporating multiple text clustering techniques and is evaluated qualitatively and quantitively.

Their process can be divided into four steps: cleansing the tickets, preprocessing the ticket texts, clustering tickets using Lingo (Osinski et al., 2004) and then further grouping the tickets using their novel hierarchical n-gram based clustering technique and finally merging similar clusters.

Mani et al. (2014) also applied the algorithm in two real case scenarios and evaluated the usefulness of the algorithm in practice. They observed that project teams used the identified clusters to find the most occurring problems in order to focus their attention on those problems. In another case, the software maintenance had been transferred over to a new service provider, and the knowledge of the repetitive problem patterns helped the new team to come up to speed quickly. Furthermore, they note that exploring clusters beyond cluster size, for instance, resolution time, SLA adherence could provide great business insights. 2 datasets: one of 1084 tickets and one of 80787 tickets.

(4) Vlasov et al. (2017) designed an AI user support system for a large Russian company. Their system can be divided into three main processes: a request classifier, a causes generation database and an answer merging process. For each of the three processes, they make use of a database in which the respecting data is stored separately.

Their problem classification algorithm is the most interesting for this research, so this will be focused on. For text pre-processing, they applied conversion to lowercase, deletion of whitespaces, number, and punctuation. Also, they deleted stopwords and reduced words to their word stems and base form (stemming). When this was done they used n-gram retrieval to find contiguous sequences. The text mining process was ended with the unification of synonymic constructions. For this process they identified three types of synonyms, namely: acronym expansions: “RFS” – “request for supply”, synonyms in the sense of the Russian language: “storekeeper” – “warehouse manager” and synonymous words in the context of SAP: “budget indicator red” – “insufficient budget”. The specification of the synonyms was done manually. For the clustering, TF-IDF was attempted but found not useful as specific words for small classes remained invaluable. Instead, they applied TF-SLF. This method is based on the fact that the term is important within a category if it occurs in most documents of this category. Finally, clustering algorithms were applied and tested. SVM and MaxEntropy appeared most useful over Naïve Bayes and K-nearest neighbors' algorithm. They use a test sample of 12554 tickets.

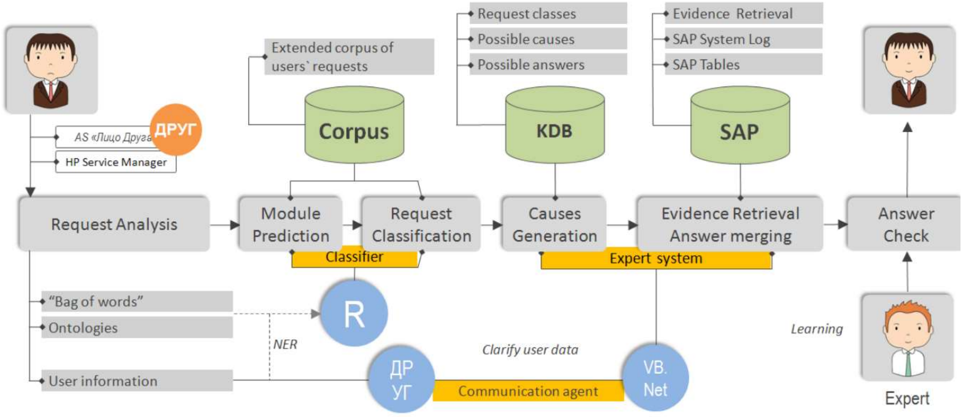


Figure 8: System architecture (Vladimir, Victoria, Marat, & Sergey, 2017)

(5) In Jan et al. (2014) a concept annotation system for tickets in IT service desk management is proposed. Their method consists of first generating n-gram phrases for which they use predefined POS patterns. To their mentioning, this methodology works very effectively for cleaning up n-gram phrases. Next, they determine the most suitable phrase using a formula consisting of different algorithmic likelihood scores of phrases. The resulting phrase is then used as a topic model and along with all other phrases clustered using Latent Dirichlet Allocation (LDA) as well as pLSA (Probabilistic Latent semantic analysis). According to (Jan et al., 2014), LDA is different from LSA in that "LSA assumes that the model parameters are fixed and unknown; while LDA places additional a priori constraint on the model parameters, i.e., thinking of them as random variables that follow Dirichlet distributions.". Their results show that both LDA and pLSA perform better than Lingo does. Two sets of 20k tickets each.

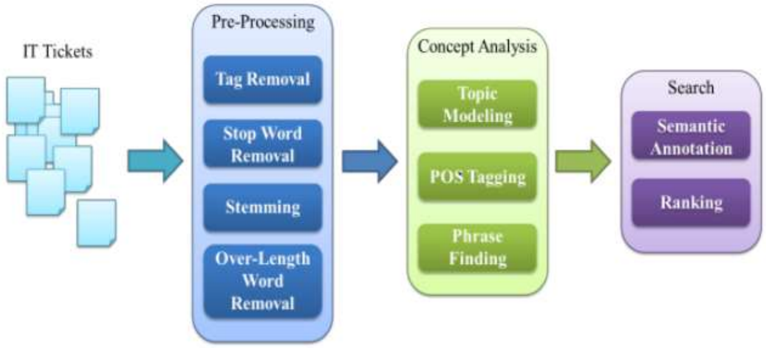


Figure 9: System architecture (Jan et al., 2014)

(6) In Potharaju & Nita-rotaru (2013) a system is designed to automatically analyze natural language text in network trouble tickets. Their case is a large cloud provider of whom they analyze 10k tickets. The system focuses on inferring three key features: (1) Problem symptoms indicating what problem occurred, (2) Troubleshooting activities describing the diagnostic steps, and (3) Resolution actions denoting the fix applied to mitigate the problem.

The problem tickets used in this research consist mostly of longer textual form. Therefore the methodology starts with hot sentence extraction. Next, a number of filters is applied in order to extract the important domain-specific patterns: Phrase length/frequency filter, Part of Speech filter and an

Entropy filter. The phrase length/frequency filter builds on the idea that important phrases often appear often and are short in length. The POS filter builds on research of Justeson et al. in which was found that technical phrases can often be placed in one of seven patterns. Each sentence is then tagged with a fitting pos tagger, and if the pos patterns coincide with one of the seven patterns, the sentence is accepted. The third patterns used information theory algorithm to calculate the information richness of sentences using Mutual Information theory and Residual Inverse Document frequency. Next to finding information-rich sentences there was also built an ontology in order to infer the lexical meaning of words.

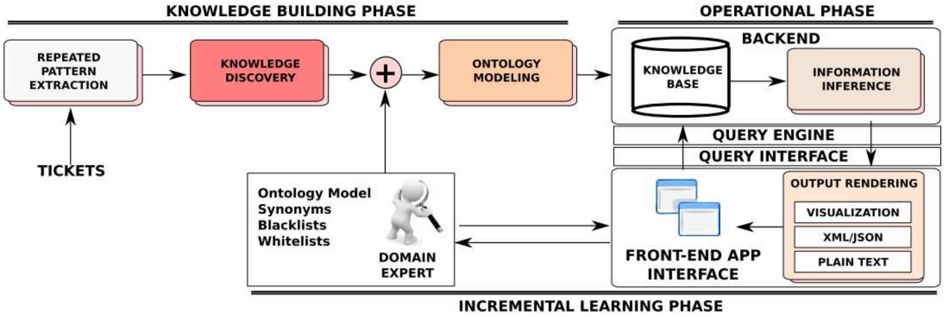


Figure 10: NetSieve system architecture (Potharaju & Nita-rotaru, 2013)

Something unique but useful that is part of their report is that they provide a chapter with challenges, indicating the challenges that they are confronted with.

2.5.1 Summary of articles

In this paragraph, the points of interests of the articles in chapter 2.4 to this research are summarized.

One large insight is that the datasets are small relative to the dataset of SSC-ICT. The largest dataset used in the articles is 80.000 tickets, less than half of the number of tickets of this research, others are mostly 20.000 tickets or even less. However, the datasets from the articles also consist of fewer categories, and they identify relatively few problems, 130 at the most, this to an expected 500 problems from this research. So even though the dataset of SSC-ICT is larger, the variety is also higher. The implication of this is that per category the number of tickets does not differ that much. Therefore, similar techniques as those used in the articles may be useful for this research. This, however, does not count for manual techniques like labeling and categorizing; it becomes more demanding when variety and scale increases.

Another insight is the clustering techniques that are used. In 4 out of 6 articles, POS patterns are extracted from sentences in order to identify problems. Furthermore, Jan et al. (2014) apply LDA topic modeling (see), and a couple of articles use Lingo's LSA clustering methodology (see).

Furthermore it can be concluded that synonyms are essential aspects of these systems. Agarwal et al. (2017) determine synonyms using their entity-operation similarity algorithm. This is a custom algorithm that calculates entity similarity based on familiar operators. Vlasov et al. (2017) differentiate three types of synonyms: acronym expansions, language-specific synonyms and domain-specific synonyms which they then manually identified.

Then, the topic of reinforcement learning within this topic was implemented only once in all six articles. P. Dhoolia et al. (2017) used customer feedback for optimizing nearly all system components, of which a domain expert first checked the adaptations.

Another recurrent component is detecting action/hot phrases; this is important when tickets consist of large pieces of text.

2.6 The ticket data

In this chapter, the ticket data that will be used is described. This is the data understanding dimension. At the end of this chapter, the dataset is compared to the datasets of comparable research and characteristics of the SSC-ICT dataset are identified as well as implications for designing the system.

Currently, all tickets of SSC-ICT are divided into two TopDesk systems. One for the Ministry of External Affairs and one for the other ministries that SSC-ICT administers. This is the case since February 2018. Before, SSC-ICT had four systems.

For this reason, the ticket data that will be used for this research will be the dataset from the start of February 2018 till the 31st of December 2018. This is a dataset of 340.000 tickets. See Appendix C for a practitioner's perspective on the tickets in the TopDesk system. See table x for an impression of a ticket and its respective fields.

TicketID	Short description	Category	Sub-category	Ticket type	Entry type	Practitioners group	Action
xxxxx	Outlook ontvangt geen mail	Applicaties	Basis	Incident	Telefonisch	S-GOS-Servicedesk	12-02-2018 10:31 lastname, firstname: Via credential manager oude wachtwoorden weggehaald. Outlook werkt weer.

Table 1: An example of an incident ticket of SSC-ICT

2.6.1 Ticket fields

The tickets have a large number of fields (40+). However, most are redundant or remain unused by the customer support operators and are therefore empty. The relevant fields are the following:

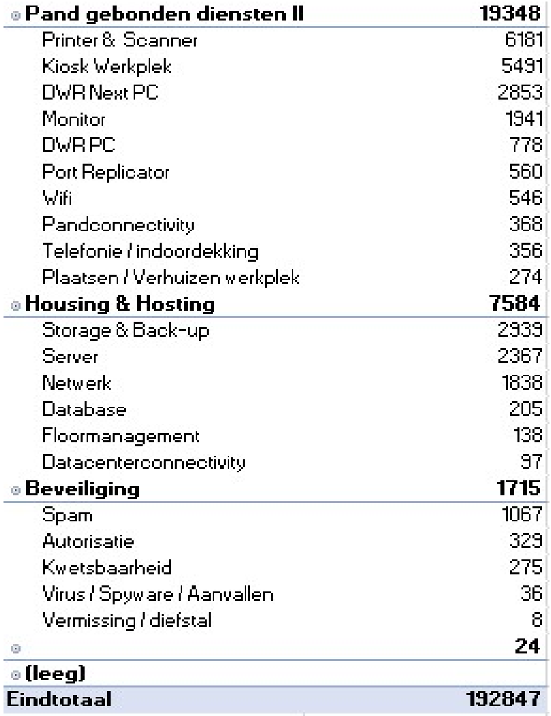
Field	
Ticket id	A unique id for each ticket, automatically generated
Short description	A summary of the ticket problem, written by the service desk operator
Request	The full description of the ticket, in case of an e-mail, the full e-mail is displayed here. In other cases, it is similar to a short description
Action	A summary of the action that follows upon the tickets, it is written by the operator.
Type of ticket	Type of customer request, either (in order of frequency): incident, request for service, internal management notification, request for information, security incident, SCOM (a monitoring system), complaint. The operator picks these.

Category	The highest level of categorization: User-bound services, Applications, Premise-bound services, Housing & hosting, Security.
Sub-category	The second level of categorization. Each of the main categories has at least five subcategories. In total there are 42 sub-categories. 50% of the tickets are covered by three subcategories. See figure x.
Practitioners group	This is the division that solved the ticket, 85% of the ticket has the service desk as practitioners group, the other tickets amongst about 300 small groups.
Entryp	Means by which customer contacted the service desk upon creation of the ticket, either telephone, e-mail, physical service desk, portal, website, manually.

Table 2: SSC-ICT relevant ticket fields

Of these fields, we further determine which of them are relevant for this research project. After data analysis, we concluded that short description and the action field are the primary resources for the project. The request field appeared too inconsistent for use. Only in the case of tickets generated by e-mails, there would occasionally be more information provided than in the short description, but it would be among much unuseful information (noise) as well. We, therefore, chose for the sake of simplicity to keep the request field out of the scope. We also decided to keep the category and subcategory fields out of this research scope. We decided this because the categorization is not problem-focused but rather organization-focused. The same problems can and do -after data analysis- occur in different sub-categories. This is not useful for intent identification.

Furthermore, data analysis showed that 30% of the tickets are categorized in the wrong sub-category. We chose not to make this inaccuracy influence our system. The remaining fields we chose to use for optimizing the training set, this is described in the next paragraph.



Pand gebonden diensten II	**19348**
Printer & Scanner	6181
Kiosk Werkplek	5491
DWR Next PC	2853
Monitor	1941
DWR PC	778
Port Replicator	560
Wifi	546
Pandconnectivity	368
Telefonie / indoordekking	356
Plaatsen / Verhuizen werkplek	274
Housing & Hosting	**7584**
Storage & Back-up	2939
Server	2367
Netwerk	1838
Database	205
Floormanagement	138
Datacenterconnectivity	97
Beveiliging	**1715**
Spam	1067
Autorisatie	329
Kwetsbaarheid	275
Virus / Spyware / Aanvallen	36
Vermissing / diefstal	8
	24
(leeg)	
Eindtotaal	**192847**

Figure 11: Ticket division by category and subcategory

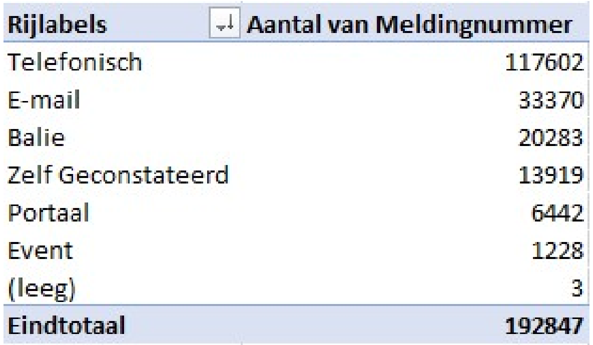
2.6.2 Data selection

In total the dataset from February to end December comprises of 340.000 tickets. After selection, 210.000 tickets remain. First, we focused on all first line tickets; with this step, we remove 40.000 tickets. Then, we chose to include only the following types of tickets: incidents, requests for service and requests for information. The other ticket types had not much to do with customers and were generally generic.

2.6.3 The input process

In this paragraph, we describe the way that tickets are registered. This provides contextual information from which we conclude some things.

Down below an overview of the division of the tickets for the different entry types: by phone (telefonisch), e-mail, physical service desk (balie), registered by user themselves (zelf geconstateerd), SSC-ICT web portal (portal) and automatically registered on event (Event).



Rijlabels	Aantal van Meldingnummer
Telefonisch	117602
E-mail	33370
Balie	20283
Zelf Geconstateerd	13919
Portaal	6442
Event	1228
(leeg)	3
Eindtotaal	**192847**

Figure 12: Ticket division by entry-type

Al tickets from all entry-types are stored in the same system in the same format and in the same database. In total, a ticket has about 40 fields that are generated (e.g., timestamp), filled in from a list of options, or typed manually. The fields that can be filled in from a list of options are the following: entry-type, category, subcategory, state, practitioners' group. The entry-type is mentioned in figure 12. The practitioner groups are the functional groups within SSC-ICT that can find a solution to a ticket. In all cases of ticket solving, as is explained by the two service desk managers that are interviewed, initially the operators try to answer the tickets themselves if they cannot find the solution, they will generate a second-line ticket that is passed on to the practitioner group that is most likely to solve the ticket, this happens in 15% of all tickets, the first line operators solve 85% of the tickets.

Fields of potential interest that are generated automatically in TopDesk are timestamp and throughput-time. Other generated fields are either not used or complementary to mentioned fields.

The manually filled-in fields are a short description, request, and action. In the short description, the ticket problem is described in one sentence. In the request field further context regarding the ticket can be provided, and in the action field, the action taken on solving the ticket is described.

2.7 System characteristics

From the related works we identify differences among the articles that impact the way the systems are built. In this paragraph, we describe these differences, how they are identified, how they are at SSC-ICT and their implication of the system. In table 3 an overview of the characteristics is shown, after that they are described in more detail.

Characteristic	**SSC-ICT**	**Implication**
Language	Dutch	- Limited availability of software/applications.
Size of dataset	Large	- Limited efficiency of manual processes.
Length of documents	Short	- Topic modeling is less useful.
Variation in intents	High	- Not suitable for topic modeling.
Variation in domains	High	- Advanced categorization
The speed of structural change in topics	High	- Minimize the need for manual work
Amount of future development	Low	- System results should be directly useful
Amount of manual work availability	Low	- Minimize manual work
Amount of potential users	High	- The potential for user feedback; reinforcement learning
Privacy restrictions	High	- Remove names from text

Table 3: QA-pair system characteristics

Language

We identify language as the language in which the tickets are written. From the related works, we see that most articles managed English systems. SSC-ICT's tickets however are all written in the Dutch language; this impacts the research in some ways. The most impactful one is that specific algorithms like POS Taggers or synonym detection techniques are trained on English datasets. They are therefore not useful for this research. A challenge, therefore, is to find accurate Dutch software.

Size of Datasets

One significant insight is that the datasets are small relative to the dataset of SSC-ICT. The largest dataset used in the articles is 80.000 tickets, less than half of the number of this research′s tickets, others are mostly 20.000 tickets or even less. However, the datasets from the articles also consist of fewer categories, and they identify relatively few problems: 130 at the most, to an expected 500 problems from this research. The implication of this is that per category the number of tickets does not differ that much. Therefore similar techniques as those used in the articles may be useful for this research. This, however, does not count for manual techniques like labeling and categorizing; they become more demanding when variety and scale increases.

Length of documents

We see a difference in length of documents between the articles with an accompanied difference in of choice of techniques. Potharaju et al. (2015) manage documents with multiple sentences; they tackle this problem by first identifying the useful phrases. Furthermore, from online research, we found that topic modeling (LDA) is especially useful for documents with multiple phrases. For short phrases, POS pattern techniques are used by the related works.

SSC-ICT's short descriptions are short phrases of on average 4,5 words long, which is short. Their action fields, however, consist of one to multiple phrases and even multiple documents like a conversation from one operator to another regarding a ticket solution. The implication is that POS pattern techniques should probably be used for the short description. For the action fields, a process of hot phrase extraction could be useful; however, this is not very accurate and should only be chosen if longer documents can for some reason not be used for action recommendation.

Number of intents

The related works all identify a small number of intents from their datasets. The largest amount is 130 intents. For SSC-ICT we expect to find over 1000 different problems, which is far beyond the number of related works. The implication for this is that manual work and correction should be minimized, at the cost of system accuracy; this impacts the choice of techniques for synonym detection as in most related works, these are identified manually.

The speed of structural change in topics

We did not identify this characteristic from the related work. However, we think it is an essential characteristic of this research because SSC-ICT has an environment that changes quickly, relative to other organizations. The implication for the system of this characteristic is that the system must be as scalable as possible, that it requires little effort to rerun the system and extract new intents.

Future development

Future development is regarding the degree to which the research's results is an actual end-product or instead, a product in continuous development. From the articles, we identified multiple different stages. For instance, Dhoolia et al. (2017) built the system with the purpose for bootstrapping an advanced cognitive system, Potharaju & Nita-rotaru (2013), Vlasov et al. (2017) built an end-product, Mani et al. (2014) and Agarwal et al. (2017) built a first-version with the purpose of applying improvements in the future. For SSC-ICT, future development depends on the results of the system. This implicates that a research result like that of Mani et al. (2014) and Agarwal et al. (2017) is needed.

Availability of maintenance

What we see from the related works is that in multiple processes manual work is used to improve the accuracy of the system or to improve the evaluation measures. In other cases, like Jan et al. (2014), was mentioned that due to limited resources manual labeling could not be performed. We, therefore, conclude that the availability of maintenance of the system is a characteristic that impacts the way a system is designed. For SSC-ICT is the case that at least for this research results the maintenance requirements should be limited and that on research following up on this research there would potentially be made more resources available.

Amount of potential users

The amount of users impacts the opportunities of gathering feedback which can, in turn, be used to improve the system using reinforcement learning. When there is too little potential for enough amount of feedback, implementing reinforcement learning would be a waste of resources, because for reinforcement learning counts: the more data, the better. On the other hand, when there is enough potential feedback, the system can benefit from this. From the related works, only Dhoolia et al. (2017) make use of user feedback to improve the system. They also happen to have the most extensive research case with a company with 450 factories and operating in 190 countries. For SSC-ICT also counts that the amount of potential of feedback is vast with over 40.000 customers. We, therefore, choose to start with reinforcement learning. However, for the first stages of the system, we should focus on the operators of the central service desk as being the users.

Privacy restrictions

Privacy restrictions is not a characteristic that we implied from the related works; however, we think it is an essential aspect for building a closed-domain system, which QA-pair system mostly are (Vlasov et al., 2017). Especially in the case of SSC-ICT, that is, a public organization, privacy is very relevant. The implications for this characteristic is that techniques by which names are filtered out of the system's results should be implemented. Moreover, that thresholds for chances of the occurring of privacy-related items in system's results need to be set.

2.8 Summary

The SSC-ICT dataset contains 340.000 tickets. The short description field and the action field contain all the information necessary for the AI components. Furthermore, we conclude that the categorization of SSC-ICT is not useful for this research. For one, it is organization centered instead of problem-focused, which we believe is not useful for intent identification. Secondly, the accuracy of the manual registration is very low with 33%; we do not want this inaccuracy to influence the performance of our system's results. However, we also see that compared to the systems of the related works, we are handling a dataset in this research with a very high variety of topics. We believe we do need initial high-level clustering, in order to go deep into the intent identification. For this reason, we choose to add a component called categorical clustering.

Regarding Root Cause Analysis, this component requires structural background information that is not available in the data. Examples of this are certain operations that led to the cause of the problem.

.

3 High-level architecture

In this chapter the high-level architecture required for the system, based on the requirements, the data characteristics and literature research, is proposed. It is decided to build a system that can be divided into three subsystems: intent identification, resolution recommendation, and reinforcement learning (see figure 13).

The system will be trained on a large dataset and applicable to new datasets or smaller subsets of data. The process for building and training the system is described in this chapter.

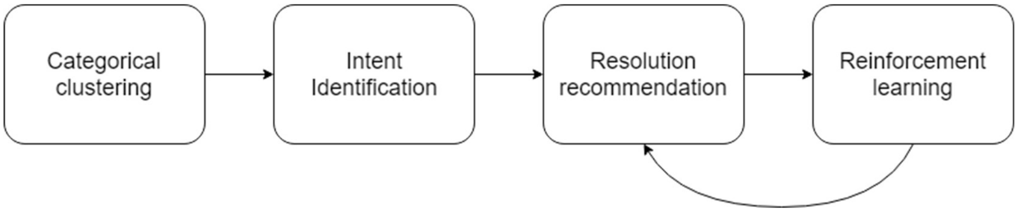


Figure 13: high-level system architecture

3.1 Categorical clustering

First, the tickets need to be ordered on categories. We decided this because detecting intents right away led to very inconsistent and noisy clusters. For detecting main clusters, there are some possibilities to be applied: keyword based-clusters (supervised), word-embedding based clustering, topic-based clustering. We see that overall, topics are very easily identifiable from the tickets based on recurring keywords like Blackberry, Outlook, and Printer.

For this reason, it is best to apply either keyword or word-embedding based categorization, as these profit most from these recurring (single) keywords. The downside to keyword-based categorization is that unimportant words like operations or adjectives may also be identified as clusters as these words are common even though they do not have a highly added value. Categorization using word-embeddings, or LSA, is the best and chosen method for this process, as it can really benefit from the single keyword categories and it excludes low-informative words automatically.

3.2 Intent Identification

Intent identification or problem identification is the process in which specific problems are identified from tickets. This can be done in a supervised methodology in which intents are identified beforehand, and new tickets are classified based on one of these intents or in an unsupervised way in which topics are created using either POS patterns in tickets or from topical word embeddings.

3.2.1 Supervised

Supervised intent identification is best applied for a closed environment. It makes use of ontologies. IT is rule-based and best applied for datasets with little variation and a constant environment, as in that the content of tickets does not change rapidly over time. This is because ontologies need to be created largely manually and will need to be manually adapted to new environments. A downside is that the input needs no be updated continuously, which is a very tough task in the case of SSC-ICT due to its scale.

3.2.2 Unsupervised

Unsupervised methodologies for intent identification are mostly either word embeddings (LDA/LSA) or patterns in word or POS forms.

Word embedding technologies are best used for longer pieces of text and very large text corpora (1.000.000+ documents), this methodology is also very fast. POS patterns work best on short pieces of text and take relatively long to process, for why they are better suited for smaller but still relatively sizeable text corpora (100 – 100.000 documents). However, for this research's system, it does not matter that much whether the processing either takes hours or minutes, as for its research goal, there is no need for processing continuously.

LDA

Jan, Chen, and Ide (2014) describe the high accuracy of topic modeling for intent identification, compared to LSI techniques. Furthermore, from conversations with data science companies was concluded that they are also working with topic modeling in numerous text clustering cases. The processed documents are however always larger than the short descriptions of the SSC-ICT dataset, and LDA performs best on larger documents.

POS Patterns

POS-Patterns are applied in four out of six of the reviewed related articles. POS patterns are in all cases a combination of a form of a verb (past, present etcetera) to that of either a noun, proper noun or adjective. The patterns are the order in which they occur and the number of nouns or adjectives.

3.3 Resolution recommendation

Resolution recommendation, action recommendation, regarding the A in Q&A, is the process of identifying actions from resolutions texts. This process is different from intent identification for some reasons. For one, resolution texts are often much longer than problem descriptions, they contain multiple sentences instead of just one. Furthermore, resolutions often consist of multiple steps instead of containing just one problem.

3.4 Reinforcement learning

Reinforcement learning or feedback learning, regarding the & in Q&A, it is the process of increasing the accuracy of the system based on user feedback. Intents can contain multiple probable actions. Reinforcement feedback helps in finding the correct action for a specific intent. User feedback will act as being the assessor on the accuracy of the action recommendation of the system. This assessment can then be used to classify the action as relevant or irrelevant to the intent based on which new intents can be solved better.

What needs to be decided is what feedback mechanisms are used to gather feedback. This depends on the type of application in which the Q&A system is applied. Examples of feedback mechanisms are amount of clicks on a specific action, a like/dislike option or search history as well as others. Combinations are also possible.

What also needs to be decided is what parameters are changed based on the feedback. Examples are looking for certain words that consistently occur in an intent with a specific action. Neural networks work very well for this process, as they find the parameters themselves. Only needs to be decided what input should be delivered to them. However neural networks work like a backbox so in many cases their inner workings cannot be evaluated. The only way to control them is to have accurate measures for their output which will have to be decided on as well.

3.5 Expected results

The in this chapter explained system outputs QA-pairs. However, because the system is composed of multiple different processes, it is reasoned that it also produces multiple results that combined produce QA-pairs. We believe that in order for the performance of the system to be measured accurately, not only the end-result should be evaluated, but the processes as well. Another argument for splitting the system's results up in its processes is due to its practical use. Categorical clusters, for instance, are a valuable resource for SSC-ICT's analytics division. Synonyms can potentially be used to create an SSC-ICT ontology which could be helpful for numerous reasons and intents could be used for more advanced business analytics. Optimizing these processes apart from each other and not only their aggregate function will benefit SSC-ICT's future potential use of these individual processes.

The system's results are split up in the following sections:

- Categorical clusters
- Sub-level clusters (intents)
- Set of actions per intent
- Front-end application

4 Performance measurement

In order to provide evidence of the effectiveness of chosen solutions and components, the system's performance will be measured and evaluated. For this research a component evaluation methodology is chosen in contrast to end-to-end evaluation, combined with both formative and summative evaluation methods as well as both automatic and manual (Resnik & Lin, 2010). A component evaluation methodology is a way of evaluating not only the end-result of the system but also its components individually. Component-based evaluation is decided for because the components are very different and the system is build in phases which are based on its components. Formative evaluation is an evaluation method that tends to be lightweight (so as to support rapid evaluation) and iterative (so that feedback can be subsequently incorporated to improve the system).

In contrast, summative evaluations are typically conducted once a system is complete (or has reached a major milestone in its development). They are intended to assess whether the intended goals of the system have been achieved (Resnik & Lin, 2010). For this research, formative evaluation is applied in all cases in which it is possible as it greatly increases development speed. In other cases, summative evaluation is applied.

Furthermore, there is a spectrum between automatic and manual evaluation. With automatic evaluation, performance can be found using custom scripts rather than manual evaluation. The same as for formative/summative evaluation counts for this, when automatic evaluation is possible and deemed faster, it is chosen.

For each of the components, unique measurements will be presented. Due to the complexity of NLP systems' output, measurements are almost always unique to their case (Paroubek et al., 2010; Resnik & Lin, 2010). In this research for each of the measurements will be explained why they are chosen.

Due to that evaluation methods are not described in the literature for QA-pair generation, the metrics are made up for this system.

4.1 Evaluation metrics

The system has two dimensions of characteristics. First the accuracy of the clustering: do tickets belong in this (sub)cluster, and two: does the cluster describe an accurate subject? Whether it is either a category or an intent; are these right and useful?

4.1.1 Categorical clusters

The high-level clusters are partially assessed manually with the help of a field expert. We chose this method due to the complexity of evaluating the accuracy of labels, and due to that there is only a relatively small number of high-level clusters and that categorization only needs to be repeated ever so often, for why it costs little time. The field expert has to decide whether the cluster labels that the system identifies are unique, value adding and not hierarchically dependent on another cluster. We implement the results into the system and re-evaluate the new resulting clusters. This re-evaluation is done using the minimal cluster size threshold, the number of clusters and the percentage of tickets clustered. These three measures are correlated. The smaller the minimal cluster size; the higher the number of clusters and the larger the percentage of tickets clustered. At some point in this process, the system will start recommending low-informative labels for categories. At this point, the limit for minimal cluster-size needs to be set.

Categorical cluster evaluation metrics
Labels:
Unique
Value adding
Hierarchically independent

Results:
Number of categories
Percentage of tickets clustered
Minimal category size

Figure 14: Categorical cluster evaluation metrics

4.1.2 Intent identification

The intents identification process is the most decisive and time-consuming part of the system regarding the accuracy of the system's results. It is also the hardest component to evaluate due to the subjectiveness of the intents. Intents are not either good or bad; there is a whole spectrum between that. Clusters may consist of some items that should not be part of them; a cluster may, in fact, better be split into two separate clusters; a cluster may be synonymous to another cluster. Due to this high complexity, determining accurate measurements is crucial.

Jan et al. (2014) use the Dunn Index and Davies-Bouldin Index, which are intrinsic evaluation methods. They calculated the inter-cluster similarity. However, this is a very minimal approach for natural language cluster evaluation due to that there are very few automatic features for similarity (their features are actually the same as the algorithm that they are testing it on, which is very dubious). Their results are also very inconsistent with findings from this research, regarding LDA. They also mention that they do not have the resources for manual evaluation or labeling, which indicates they would have used these methods otherwise.

Internal and External cluster evaluation

Cluster evaluation is divided into two groups: internal evaluation and external evaluation. They differ in whether or not external information is used to validate the goodness of the partitions (Liu, Li, Xiong, Gao, & Wu, 2010). For internal cluster evaluation thus only internal features of clusters are used.

We believe this is not a very accurate method to determine whether intents are actually unique and specific, as for these measures external information is needed.

External evaluation, however, generally relies on a predefined structure. For these structures, accurate labels are needed. Moreover, labels we do not have and do not want to have since it limits the dynamism of the system. Manually labeling clusters is much work when we expect to identify up to a 1000 different intents. For this reason, we came up with a new cluster evaluation methodology.

Custom evaluation methodology

We create a golden evaluation set that is manually created by some field experts and evaluated multiple times. Then, we compare the items that are found in the cluster of the system and that of the golden set to each other. We calculate for each ticket which tickets are in the system's parent cluster compared to which tickets are in the cluster of the golden set, divided by the sum of the number of tickets in the cluster of the golden set and the system's set divided by two. We then sum up the scores of each of the tickets and divide it by the total amount of tickets.

Due to that, we divide the mutual ticket count by the average of the two cluster sizes we avoid the problems that occur when creating a cluster set of a unique cluster for each ticket or one large cluster with all of them. The tickets that are in a large cluster in a golden set would get a very low score due to that. On the other hand, the clustering problem of clustering all tickets in one big cluster also gets a low score due to that the score is divided by the number of tickets in the system's cluster. The resulting score is then the average percentage of mutual tickets in a cluster for each ticket on a range of 0 to 1. A score of 0.5 for the system means that

We determine the minimal quality score to be the scores for both the case of generating all unique clusters or that of all tickets in the same cluster. Random assigning of tickets to clusters leads to scores that are almost always lower than those.

Manual creation golden test set

Regarding manual evaluation cluster evaluation; we identify three options: manually evaluating all ticket and clusters, manually evaluating a sample of tickets, or using a golden evaluation set.

Manually evaluating all tickets and clusters is an option in case the amount of tickets is low, the amount of clusters is high, and the amount of evaluation iterations is low as well. Manually evaluating a sample set of the results is useful in the case that the amount of tickets is high, the variation is low, and the amount of design iterations is relatively low. When using a golden evaluation set, a sample of the tickets is clustered most optimally, manually. This set is then compared to extracts of a system's cluster results using a multitude of different algorithms. Using a golden evaluation set is chosen due to its use for large amounts of tickets, high variation and a large number of evaluation iterations. It is applied by making three field experts of SSC-ICT cluster 333 tickets from 3 cluster categories, totaling to 1000 tickets, manually. It is chosen to select samples from categories and not from the whole dataset because in the second case there would be too many clusters that would consist of 1 ticket, which is useless to evaluate since only the clusters that contain multiple tickets are relevant.

Furthermore, it is decided to use multiple categories instead of one due to the differences between the categories. Some are larger; some contain a relatively high amount of intents; some consist of very concise short descriptions. We chose three categories; a large one, a medium sized one and a small one with around 333 tickets so that the evaluation covers it fully. The topics are also variative, one major subject, one application, and one small service.

Coverage of the system means the number of tickets that are successfully combined in subclusters to the total amount of tickets. A threshold is used to minimize the number of small clusters for that the really small clusters are of little use to the system. The threshold parameter will have a very high influence on the coverage rate, as potentially every ticket can be clustered in an intent of its own which results in a 100% coverage rate with a threshold of 1, so the chosen threshold has to be provided with the coverage rate. This measure is objective, it can be directly inducted from the system's results, so there is no need for a domain expert.

Processing speed is the speed of the system. Practically, this can be either the speed of processing one ticket to recommend action or the time it takes to process the whole dataset, in order to train the system. The last one is chosen as the metric as this gives the most accurate results. In general, this evaluation metric is not critical in case it stays under about 10 hours, as the system does not need to be updated daily.

Coverage
Custom evaluation score
Processing speed

4.1.3 Set of actions per intent

As is concluded in the chapter Data Understanding, only a small portion of the actions contain valuable information for the system.

The challenge in the actions is to filter out irrelevant actions, of which there are many, and to keep thus only the actions that are relevant to the intent. The measure will, therefore, be the number of useful actions to the total amount of actions proposed by the system; this can be calculated by manually testing on a sample.

% of useful actions

4.1.4 System end-result

We determine the end-result of the system by combining the scores for all independent components. Only for the intent identification, we will use a new measurement due to that the measurement that we used for that is useful for comparing two techniques automatically, but not for determing the accuracy of the end-system. We will do this by manually evaluating the intents on their specificity. Specificity is the degree to which the tickets in a cluster describe in fact the same problem. We will use a 75% threshold for this. If at least 75% of the tickets belonging to in intent are about the same problem it passes. If the specificity score is lower than 75%, the cluster is deemed incorrect.

4.2 Tool selection

In this paragraph, we describe the tools that we chose for the different processes, as well as the arguments for the choice of these tools.

Building the system

We use Python as the primary programming language for building the system and most of the components and features. We made this choice because of the number of available libraries regarding Data Science of Python. Also, Python is very well suited for building systems from scratch. A disadvantage of Python to for instance R is its processing speed. However, this is not a significant problem due to the relatively small amount of data compared to other data science projects. We use the Spyder IDE from Anaconda Open Source Distribution as Integrated Development Environment for Python.

High-level clustering

Lingo3g

Initially, we chose Lingo3G for performing the high-level clustering process. This application was found from multiple scientific articles (Mani et al. (2014); Jan et al. (2014); P. Dhoolia et al. (2017)). The application uses latent semantic indexing to generate clusters of topics from a set of documents. The strength of this application is the ease with which parameters are tested and adjusted. A testing process that would otherwise take weeks now takes a couple of days. After some initial testing the results showed potential, and after adjusting the parameters of the system, the results were quickly useful. Adjusting weights for individual labels, as well as adding custom stopwords perfected the system.

Carrot2

Carrot2 is the free version of Lingo3g. In contrast to Lingo3g, carrot2 is memory based and has a limit of clustering up to 10.000 documents.

POS tagging

The big problem with finding a good POS tagger is that these applications are language-specific. The availability of Dutch POS taggers is very sparse. After a thorough search in which we compared multiple systems, we found the following two taggers which both have their advantages and disadvantages.

Frog POS Tagger

The Frog POS tagger was by far the most accurate POS Tagger, this was identified by testing the tagger on a subset of the SSC-ICT dataset and comparing the results to manually determined results. A downside of this POS Tagger is its speed and its difficulty to use. In order to use it a separate LINUX virtual machine (VM) needs to be created on which multiple large packages need to be installed and on which the Frog application can be run. This machine then needs to communicate with the Python server to send data and retrieve results. Its speed is very low relative to other POS taggers with the processing of 900 words per second. Processing all short descriptions of all SSC-ICT tickets takes therefore about 5 hours.

NLTK-Spacy tagger

The NLTK-Spacy POS tagger is much faster than the FROG tagger. About 20 times as fast. The accuracy is however much less. It is a python library and therefore easy to call. We used this POS tagger while building parts of the system in which the accuracy of the results did not matter as much.

LDA

Gensim

For the topic modelling process in the intent identification component we use the Gensim library for Python. This library is the most used Library next to the SciKit library, and we find it has the most documentation.

Lemmatization

- Frog Lemmatizer

Just like with the POS tagging, the lemmatizer of the frog system was much more accurate than other algorithms. Again, the application was much slower than other applications. We used this lemmatizer for when we evaluated results on quality. Frog does not include a stemmer.

Stemming

- NLTK-Snowball stemmer

This system was much faster than the other one and was used when the accuracy of the results did not matter as much. Stemming did not lead to better clusters than lemmatizing did.

Deep Learning scripts

Python has several options regarding solutions that use deep learning. However, the solutions from the Gensim library had by far the most use cases and document support and were up-to-date.

- Gemsin Library
    - Word2vec synonyms
    - Bi-gram model

Synonym database

The OpenDutchWordNet (ODWN) database was by far the largest open-source Dutch lexical database and acknowledged by multiple large parties, among which the NLTK library. For this reason, this database is chosen for finding ordinary Dutch synonyms.

FastText

FastText is a technique developed by Facebook in 2016. It is a very accurate classification method for small documents using neural networks. The documents require labels. They have a python API.

Custom scripts

Due to the large community (and therefore feedback and use-cases) behind Python and its ease of use in creating scripts from scratch, it was decided to use this programming language for building the custom scripts.

5 Modeling and results

This chapter first describes the chosen tools for this research and then the choice of techniques for the processes intent identification, resolution recommendation and reinforcement learning for the system.

5.1.1 Categorizational clustering

In this section we describe

The column with the short description of all tickets, along with their ticket ids, is exported from the excel dataset and converted to XML-format, this is a file of 450.000 lines. We then process this file in Lingo3G with the following custom parameters on top of the standard parameters (see table 2)

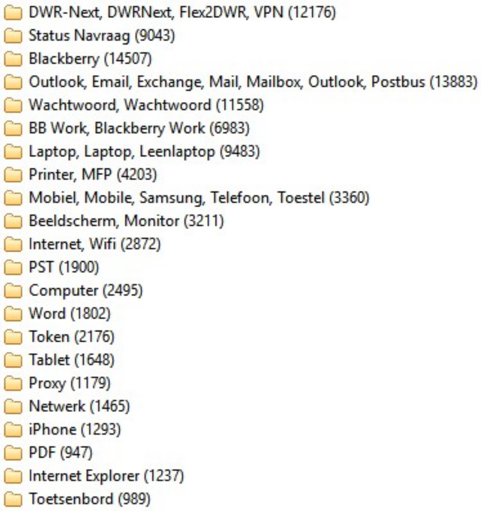
- Minimum cluster size: 0,0010%
- Cluster count base: 20
- Maximum hierarchy depth: 1
- Phrase-DF cut-off scaling: 0,20
- Word-DF cut-off scaling: 0,00
- Maximum top-level clustering passes: 8
- Default clustering language: Dutch
- Language aggregation strategy: Cluster all documents assuming the language of majority

Table 1: Parameters Lingo3G

For the categorizational clustering, three techniques are attempted based on domain research: LDA, POS Patterns and Lingo3G clustering. LDA did not show good results. The resulting clusters are overlapping.

POS patterns were also not effective. The POS patterns were too specific and did not capture the global category.

Lingo3g however, worked very well on the dataset. After having tweaked with the attribute settings, amongst other things promoting short (one-word) labels and increasing the expected number of clusters, a decent process-based ticket cluster overview came forward (see figure 6).



DWR-Next, DWRNext, Flex2DWR, VPN (12176)
Status Navraag (9043)
Blackberry (14507)
Outlook, Email, Exchange, Mail, Mailbox, Outlook, Postbus (13883)
Wachtwoord, Wachtwoord (11558)
BB Work, Blackberry Work (6983)
Laptop, Laptop, Leenlaptop (9483)
Printer, MFP (4203)
Mobiel, Mobile, Samsung, Telefoon, Toestel (3360)
Beeldscherm, Monitor (3211)
Internet, Wifi (2872)
PST (1900)
Computer (2495)
Word (1802)
Token (2176)
Tablet (1648)
Proxy (1179)
Netwerk (1465)
iPhone (1293)
PDF (947)
Internet Explorer (1237)
Toetsenbord (989)

Figure 15: 20/150 clusters from Lingo3G

5.1.2 Iteration 1: Lingo3G

Lingo3G applies LSA (Latent Semantic Analysis) using TF-IDF word embeddings on a text corpus and then applies SVD for dimensionality reduction. Its algorithm consists of multiple steps: pre-processing, frequent phrase extraction, cluster label induction and cluster content discovery. The pre-processing step removes stop words from an external list that is created by a field expert.

Furthermore, this expert also identifies synonyms and label name. Because the input consists of only one sentence, we skip the frequent phrase extraction process. The pre-processing step is supervised, as in label preference, synonyms and stop-words can be predefined. The other processes are unsupervised. As such the resulting labels are made up by the system. SSC-ICT currently has no accurate problem-based categorization of their tickets, and we believe the categorization of Lingo3G (after removal of stop words and non-relevant labels) is an accurate, specific, and data-driven representation of the problem topics within SSC-ICT.

Results

Lingo3G generates 117 clusters from the ticket data. With the largest being 10% of the whole ticket corpus and the smallest 0,05%. The ten largest clusters accumulate to 65% of the ticket corpus. 20% is not categorizable, 15% is part of the other 107 clusters. A visualization of the weighted clusters is provided in figure 7.

# of tickets: 210.000
# of clusters: 117 (can be determined manually)
% of tickets clustered: 80%
Speed: Couple of seconds

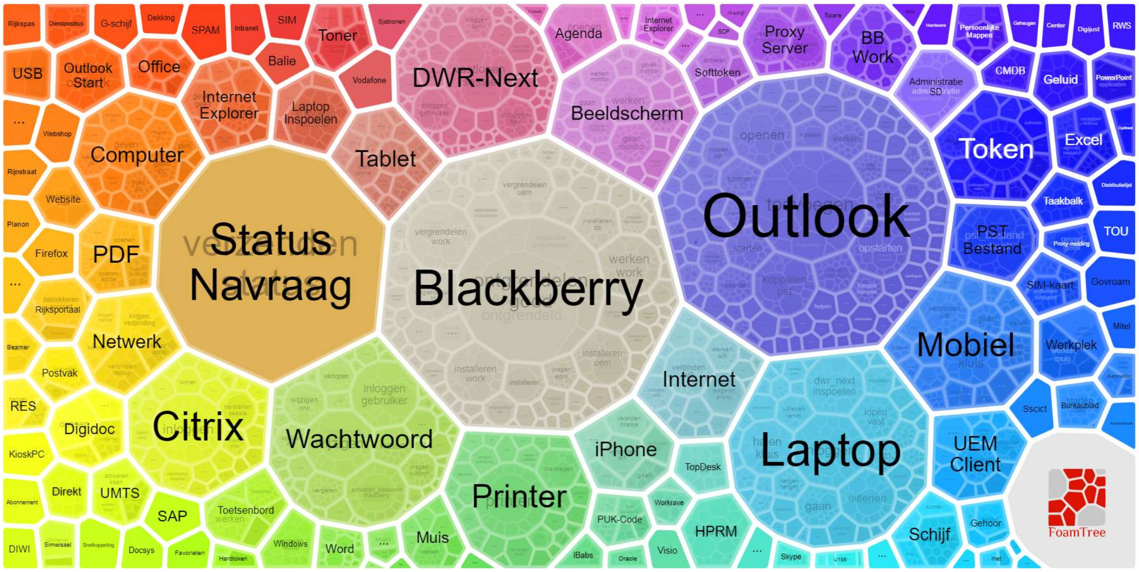


Figure 16: High-level cluster results from Lingo3G

5.1.3 Iteration 2: Carrot2 + Levenshtein distance

Since Lingo3g is not open source and after having contact with the company that owns the software it would be known that a business license is costly. For this reason, we sought a solution that could do the same but then for free. We deemed this possible due to the limited usage of lingo3G's capabilities, as the system mostly only used single word labels for categories, whereas Lingo3G is, in contrast, especially good at detecting clusters for sensemaking multi-word labels. The problem is however that no such solution exists. Therefore, we looked at the free version of Lingo, which is carrot2. The downside to this version was that it is a memory-based algorithm whereas Lingo3G works with indexes.

For this reason, only 10.000 tickets can be clustered at the same time. A solution to this was found in that we generated a random sample of 10.000 tickets from the complete dataset. We then fed this sample through the carrot2 system and extracted the clusters. Next, using a custom script, tickets were classified based on the labels of carrot2's clusters. This was done by first tokenizing the short descriptions and then searching for the cluster labels from the cluster list from top to bottom, based on the cluster size of the extracted clusters from the 10k sample. Additively, we decided that we could add the Levenshtein distance to the custom script for word labels of at least five characters (in order to prevent misclassification of the algorithm finding smaller labels like ″i.e." (internet explorer) in for instance "is" or "be"). This way typos or concatenations of word labels are also clustered, something that the Lingo3G algorithm did not always do automatically; this increased the coverage by another 10% whereas before the coverage was about the same as Lingo3G's clustering method. The custom script, however, does take some time to classify the tickets to the clusters of the carrot2 algorithm: about 30 minutes for 210.000 tickets. See Appendix X for the resulting clusters and their document count.

# of tickets: 210.000
# of clusters: 150 (can be determined manually)
Coverage: 88%
Speed: ~30 minutes

5.2 Intent-level clustering

This paragraph describes both successful and unsuccessful iterations of building the intent identification component. First, we apply POS patterns which we continue to use for intent identification. The iterations after the POS pattern iteration build on the POS pattern process, so these results are compared to the results of the POS patterns. Next, we describe the application of Topic Modelling (LDA) on the dataset and evaluate the results.

5.2.1 Iteration 1: POS patterns

For the identification of unique problems, we applied POS Patterns to the "Korte omschrijving" text. From the related works, it was clear that this was the go-to method to extract intents for short text and high variety corpus. We use the combination of operation-entity POS patterns, that is described in P Dhoolia et al. (2017). The operations are verbs; the entities are nouns and adjectives.

For preprocessing, first stopwords are removed using an online freely available stopword-list. Labels of the categories in which the tickets are classified are removed as well, to avoid redundant intent labels. Next, we tag the remaining words on Part of Speech. If a verb is detected, the system combines the nearest nouns or adjectives with them in order to form a two-word phrase. If no verb is detected the system uses the remaining words as intent. In most cases that no verb exists in sentences, the sentence is short, so that the phrase remains short. In the exception of longer phrases with no verbs, the whole sentence is ignored.

Results

We show the results in table x. The total amount of tickets that the system converts to intents is about 110.000; this is slightly more than 50% the categorized tickets. After looking at the unclustered tickets, we conclude that ignoring the sentences that have no verb and contain more than two of the nouns and adjectives is the cause of this.

Total tickets covered: **109908**

Threshold: 10
Coverage: **75955**
# of intents: 1490

Quality Scores:
Large: 0.3747
Medium: **0.4954**
Small: 0.2520
Average: 0.3740

5.2.2 Iteration 2: POS patterns: bigrams added

Bigrams

Identifying and combining bigrams makes sure important concatenations of words that are separated with spaces are not separated when POS patterns are applied. For instance, the virtual environment application “DWR Next” becomes DWR_Next. The software that we use for this is Gensim. We chose this module because it makes use of neural networks and thus can be easily and effectively trained on a training corpus. The advantage of this over database-based modules is that domain-unique words like “DWR Next” can now be identified.

In order to avoid that the bigram model combines verbs with entities as bi-grams which appeared to happen during a test run, we trained the model on a ticket dataset in which we removed all verbs. The resulting model is then stored and can be applied at any moment on any sentence to identify SSC-ICT unique bi-grams. Examples are DWR_Next, PST_bestand, ontgrendel_code, UEM_client and activation_password.

Results:

The results were not as big as expected. The coverage only went up slightly, overall, and also slightly for the intents. The quality scores went below the scores they would have without applying them. We conclude that bi-grams may look nice in the labels, which they do, but for the actual performance of the system, they provide little benefit.

Total tickets covered: **111938**

Threshold: 10
Coverage: **77414**
# of intents: 1504

Quality Scores:
Large: 0.3739
Medium: **0.4729**
Small: 0.2492
Average: 0.3653

5.2.3 Iteration 3: POS patterns: adding synonyms

The adding of synonyms is an advanced and challenging step. It is difficult because the boundary between whether words are synonyms or not is somewhat inconsistent and a grey area. Furthermore, words can have multiple meanings. However, we hypothesize that the advantages of implementing synonyms overrule the disadvantages. An advantage is increased merging of clusters, which decreases the number of redundant clusters and increases the number of useful actions per cluster.

As we described in chapter “Background”, there are two types of synonyms: general language synonyms and domain synonyms. We hypothesize that implementing domain synonyms is less risky but less rewarding as well.

Initially, the idea was to use the dutch synonyms list of lexical database OpenDutchWordnet to identify the general synonyms for the SSC-ICT corpus. However, after identifying the synonym sets for the SSC-ICT corpus using a custom script that put all the words of the SSC-ICT vocabulary against the ODWN synset, we found the resulting synsets unuseful. Most synonym sets contained words that were indeed similar but did not mean the same in the context of SSC-ICT. We, therefore, chose to use a custom Word2Vec deep learning model to devise suitable synonym sets.

First, we trained a model on the complete corpus. Before this, we lemmatized the corpus to increase normalization; this showed a positive effect in higher similarity scores for similar words when compared to unlemmatized versions of the model. Next, we split up the vocabulary in verbs and nouns/spec/adjectives, and we wrote a script that calculated the similarity score of all combinations of the words for each of the vocabularies. Then, we computed lists of words that were similar with a similarity score of at least 0.70 (on a range of 0 to 1). We sorted the lists on the frequency of the words in the vocabulary with the purpose of that the most frequent word would come first in the list. This word would be the “alfa” word by which all other similar words are replaced. The resulting synonym sets were exported to a list and manually checked. About 50% of the synonyms were accepted. In total, we identified about 100 synonym sets for entities and 30 for verbs, with on average 3-4 words per synonyms set. See Appendix E for the list of synonyms.

In order to implement the synonyms in the system we wrote a script that simply replaced the respective synonyms by their alpha synonym in the dataset that is input for the POS Tagging process.

Results

The coverage went up by 5000 (7,5%). The quality scores remained about the same. We conclude that the synonyms have a positive effect on the system, but not drastically. Increasing the number of synonyms would possibly improve the system more. In order to do this, the similarity threshold would need to be lowered, and more manual work would be needed to check them. However, the process of checking the synonyms is very fast since it is very intuitive. Checking 100 synonym sets takes about 10 minutes.

Total tickets covered: 114512

Threshold 10:
Coverage: 81052
# of intents: 1500

Quality Scores:
Large: 0.4106
Medium: 0.4844
Small: 0.2403
Average: 0.3784

5.2.4 Iteration 4: POS Patterns: multi-threaded processing

Training a new iteration of the model took about 4,5 hours. We identified the bottleneck to be the Frog POS tagger which runs on an Ubuntu Virtual Machine (VM). The developers of Frog warn on their website for the slowness of the software. However, we found a way to increase the speed of the software by more than 250% by using multiple ubuntu instances. We split up the processing script using the ThreadPool Library of Python: we wrote a script that divided the categorical clusters over the Ubuntu instances.

5.2.5 Iteration 5: Topic modeling (LDA)

Due to the high expectations of LDA in text clustering (in research but also in online communities and data science companies that we had contact with) and also the high scores of the technique in the article of Jan et al. (2014) (even though they used internal evaluation scores) we decided we had to attempt this technique. Before using the intent evaluation datasets, we first attempted to apply LDA on one complete large cluster, because LDA requires a large number of documents as input and we could immediately see the results from this and conclude whether we should continue testing the technique.

For this experiment, we used the complete dataset of the outlook cluster, which comprises about 15.000 tickets. For pre-processing, we lemmatized the dataset, and we used a dutch stopword list. Then we extract the complete vocabulary (unique word list) and convert the documents to a TF-IDF matrix using the Gensim library. We use these files as input for training the LDA model. For determining the number of topics, we tried using a widely known methodology which makes use of the perplexity score of the clustering results. However, this methodology recommends to use a maximum of 30 topics, which we find very small and the results also show very general topics. We then choose to go for 100 topics, which is a rough estimate.

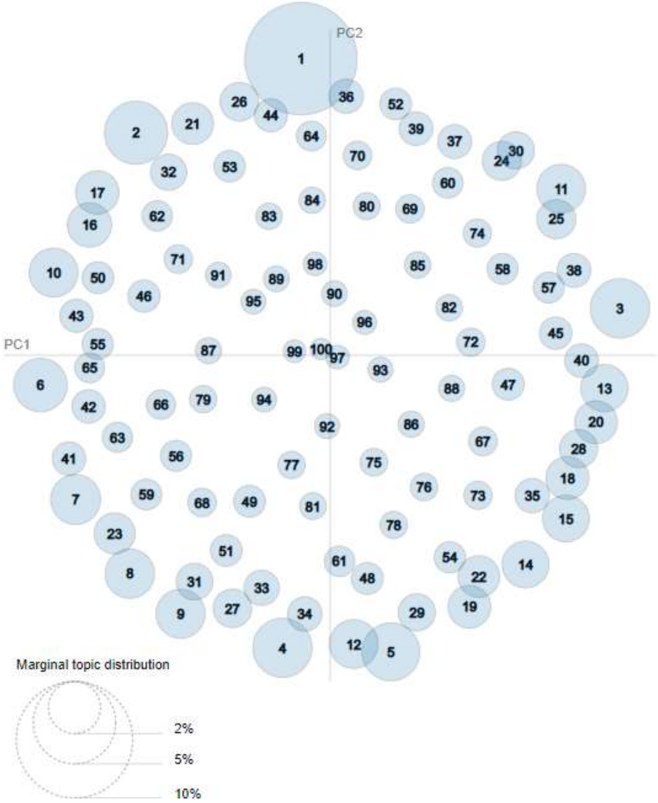


Figure 17: Visualization of LDA topic distribution of the "Outlook" category

Results

In figure 18 we show a plot of the topic distribution; this is a two-dimensional grid in which the distances between the word embedding vectors of the topically related words are visualized. When looking at the terms that each topic describes we see LDA does cluster topics indeed relatively neatly. For instance, the largest cluster, number 1, which contains about 6% of all tickets from the category describes the words “PST”, “bestand” and “koppelen”, or “pst bestand koppelen” which is indeed an intent in the outlook cluster and also the largest one. Some of the smaller topics are not correct due to certain terms that provide little informative quality.

Total tickets covered: 100%

Quality Scores:
Large(100): 0.4063
Medium(50): 0.4200
Small(25): 0.3062
Average: 0.3775

The scores are slightly lower than that of the POS tagging but still pretty good considering little preprocessing is done, and no synonyms are applied. Especially the smallest cluster scores better than on POS Tagging; we do not know why this is.

5.3 Resolution recommendation

In this section is described how the resolution recommendation process should work.

For the resolution recommendation process, we combine the tickets in the clusters with their respective actions.

Using a custom algorithm that makes use of the ratio of verbs as well as numbers in a sentence successfully removes all e-mail related noise like signature and salutation as well as TopDesk related noise consisting of the name of the operator and timestamp.

Next, we remove empty actions fields and combine double actions; this increases the weight rate that we match to these actions.

A domain expert has labeled 2.000 actions in order to identify what actions contain valuable information regarding the actual solution to the problem. 30% of the actions appear to be useful. This rate can be used to evaluate the system′s recommendation to a bottom limit. Another conclusion of this analysis is that shorter actions more often contain valuable information rather than longer action texts. For this reason, only the shorter action texts, those that contain less than 300 characters (on average three sentences), are analyzed.

5.4 Front-end application

In order to acquire feedback on the system, an application needs to be decided for and built. In chapter 2 an overview of applications of QA-pairs is provided. In this chapter is explained what application is chosen and for what reasons. Furthermore, we discuss the details of the application.

Application description

It is decided to build a customer knowledge base system primarily for use by the customers of SSC-ICT. This system provides the option for a user to type in a short description of any incident, and the system will recommend intents and actions belonging to these intents. Furthermore, it will provide the possibility for the user to provide feedback on the results. This feedback is used for evaluation as well as for use by the reinforcement learning algorithm.

Argumentation for the choice of the system

We chose this system because of the substantial benefits it can provide. It would save a significant amount of the service desk operators' work as the most straightforward tickets can be answered by the customers autonomously.

Furthermore, providing the application to the 40.000 customers of SSC-ICT comes with a large amount of feedback. This feedback can be used using reinforcement learning to improve the system further.

Application's process:

The application processes the input text live. We apply the same pre-processing to the text that we use for training the system. After that, we determine the corresponding category in the same way that the tickets are appointed to categories while training the system. Then, the input text is classified using the trained LDA model for that category. The outcome of this is a list of topics along with their contribution percentage. The topic with the highest percentage is chosen as the being the intent for the input.

Feedback mechanisms

The system provides two ways to gather feedback from users:

- Possibility to classify an intent as right or wrong (mark)
- Possibility to select actions as useful (like)

A system expert manually reviews the feedback, and if accepted it is incremented in the reinforcement learning algorithm.

5.5 Chapter conclusion

In this section, we look back at the modeling that we describe in this chapter and built conclusions for the system's design based on the results.

For the categorization component, we decide to use the Carrot2 LSI clustering implementation along with assignation of tickets to the cluster using the Levenshtein distance. The score of more than 88% coverage is an excellent score for categorization.

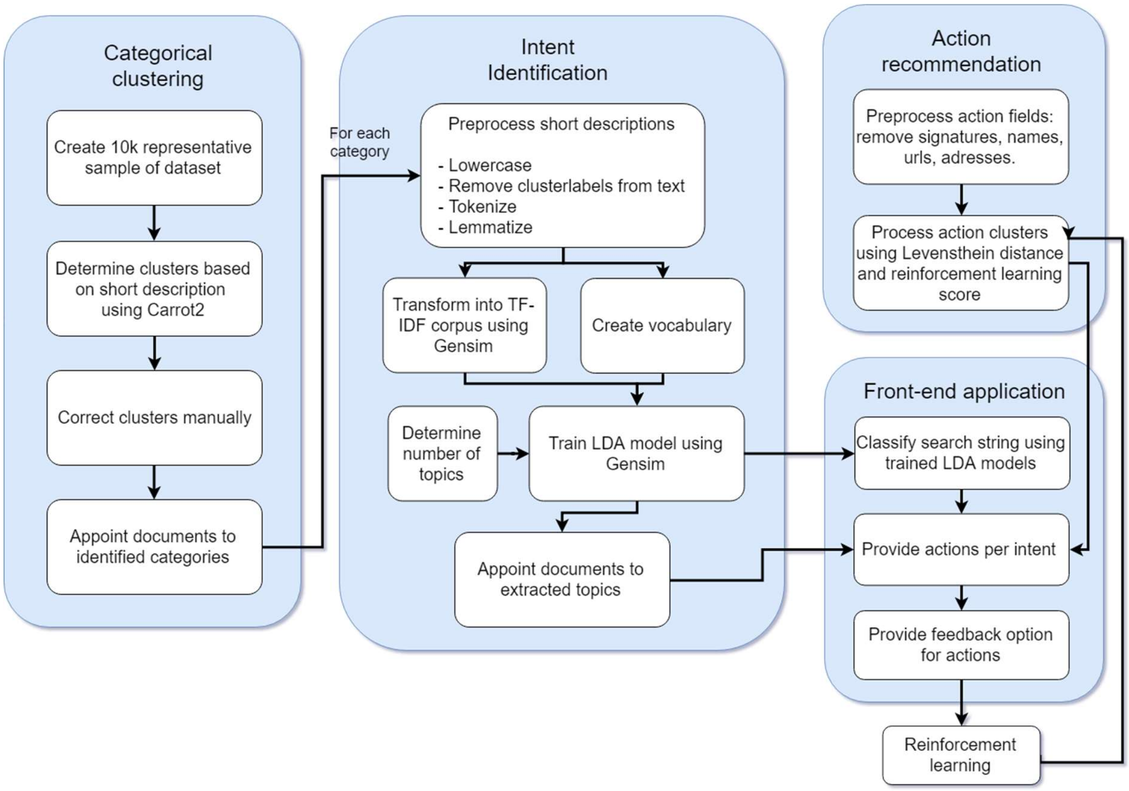
In the intent identification process, we focussed on the POS patterns that arose from the related works and LDA Topic Modeling which is a much-valued technique in the research community. Despite that all odds, in our eyes, were against LDA, we believe that LDA outperforms the POS Pattern process. The evaluation scores based on our own evaluation measure may be slightly lower than that of POS Patterns, but the coverage is much higher, as well as its processing speed, and the expected future potential improvements of LDA are much higher as well. We will describe these improvements in the next chapter.

For the action recommendation process, we propose a preprocessing methodology as well as a low-effort clustering methodology. The front-end application

We attempted multiple categorization techniques, multiple intent identification techniques, we cleaned and clustered the action field, proposed a method for searching through the clustered intents and proposed a methodology for implementing reinforcement learning.

6 The system

In this chapter, we provide an overview of the whole system. Furthermore, we describe the evaluation of the end result of the system, and we compare this to the minimal expected quality level that we set in this chapter as well. We provide an overview of the complete system in figure 19.



Categorical clustering
Create 10k representative sample of dataset
Determine clusters based on short description using Carrot2
Correct clusters manually
Appoint documents to identified categories
For each category
Intent Identification
Preprocess short descriptions
- Lowercase
- Remove clusterlabels from text
- Tokenize
- Lemmatize
Transform into TF-IDF corpus using Gensim
Create vocabulary
Determine number of topics
Train LDA model using Gensim
Appoint documents to extracted topics
Action recommendation
Preprocess action fields: remove signatures, names, urls, adresses.
Process action clusters using Levensthein distance and reinforcement learning score
Front-end application
Classify search string using trained LDA models
Provide actions per intent
Provide feedback option for actions
Reinforcement learning

Figure 18: A process view of the system

The figure in figure 19 shows the complete process of training the system and recommending actions to customer input. For training the system the categorical clustering and intent identification are used. First, the categories are determined using LSI indexing. Then, the tickets are appointed to one of around 100 categories (for the SSC-ICT dataset). After that, the intents are identified.

For each of the categories, we apply the following process. The system preprocesses the short descriptions of the tickets and the complete corpus of short descriptions for a category transformed into a TF-IDF corpus, in which the preprocessed short descriptions are the documents. Parallel to the creation of the TF-IDF corpus, the system creates a vocabulary for the category. Then, the expected amount of topics is determined and used as input along with the TF-IDF corpus and the vocabulary as input to train the LDA model. Once the system has trained the model, the tickets are appointed a dominant topic which is the intent.

The system than grabs the action fields for each of the tickets of each intent and excludes doubles and actions that are very similar using the Levenshtein distance. The result is a list of actions for each intent.

When a customer types in a problem in the front-end application, the system recommends a intent and the customer can choose an intent which he or she thinks fits best. The system then recommends a list of action on the intent. The list is sorted based on feedback of customers as well as on a score that is provided by a deep learning classifier which can distinguish completely useless actions to probable actions.

6.1 Minimal quality level

Now that the context of the system is determined we can set a minimal quality level. There is no way to base this on other research because the cases are just too different. What we can do is describe from what moment SSC-ICT would benefit from using the system.

Because effectively, the system replaces service desk operators, success on a purely business-perspective would be reached very quickly, even at a success rate of about 10%. However, the main goal of the system is to increase customer satisfaction. Being able to have the option to solve an IT incident without the need for a service desk operator, 24//7, would be of a positive influence on customer satisfaction. However, taking into account that the system is not flawless, there is a point where users might find it hindering to use. One could say to that however, that the user may simply choose to not use it, leaving it only to those that are interested or for everyone but outside of the service desks working hours. Still, the image of SSC-ICT depends on the application as it will be one of the very few things of SSC-ICT that the 40.000 customers are confronted with. However, the system might be given some slack due to it being a pilot for Artificial Intelligence. On top of that, the system will improve when feedback is applied in the right manner.

In short, setting a minimal quality is a process of pure estimation. We think a success score of at least 30% is a good starting point, and increasing it to 50% over time by improving the system and using reinforcement learning should be wanted.

6.2 Results

In chapter 5 we evaluated the components of the system independently in order to decide what technologies we recommend for these components. In this chapter, we evaluate the results of the complete system. We do this by manually determining the specificness of the resulting intents and the number of actions that we require at a minimum for useful action recommendation. The specificness is vital because when a cluster is specific, i.e., it describes only one intent, we can safely say the tickets that that intent covers are successfully clustered, and thus provide a percentual success rate of the intent-identification process.

During the process of determining the uniques we also identified clusters of tickets of which the short decriptions is too general for intent identification. The short descriptions of these tickets were generally one of the following: “problem with outlook”, “question about outlook”, “help with outlook”. We use “Outlook” as an example category but they appear for every category.

We remove these general tickets from the calculation of the success rate, this does not impact the credibility of the success rate of the intent identification because they would have been clustered if they would have been described more accurately. However, we do find them an interesting result of this research because it provides insight for SSC-ICT into what percentage of tickets are processed incorrectly, we, therefore, provide these results as well.

"Outlook" category:
Total amount of tickets:
13341
Tickets clustered in specific clusters:
8034
Number of too general tickets:
1323

Succes rate: 55,8%

"Excel" category:
Total amount of tickets:
721
Tickets clustered in specific clusters:
436
Number of too general tickets:
167

Succes rate: 48,6%

"P-Direkt" category:
Total amount of tickets:
286
Tickets clustered in specific clusters:
220
Number of too general tickets:
89

Succes rate: 66,5%

From the results can be seen that the success rate of the intent identification process is on average around 55%. This score means that, on average, the system can identify a right intent for a ticket 55% of the time. Furthermore, we conclude that between 10 and 20% of the tickets that are part of a category are described too vague to extract any meaning out of them. On top of the 12% of the categorizational clustering component (88% success rate), we say that between 20 and 30% of all tickets are described too vaguely by the operators.

In order for the system to solve 55% of the tickets, the recommended actions should be useful. In chapter 5 we describe that of all tickets, about 30% contains a useful action. Looking at the intents, which are almost always larger than 10 tickets and often larger than 100 tickets, the chance that an intent has at least one useful action is large. Furthermore, if this does not appear to be the case the action could always be added manually by an operator. So once enough feedback is received from users, the right actions are filtered from the less informative actions and the system will able to recommend a useful action to an intent most of the time.

7 Deployment

In this chapter, we describe how SSC-ICT should make use of the QA-pair generation system. We describe the first uses of the system, and what potential improvements SSC-ICT should apply in what order, in order to improve the system.

7.1 Potential usage of the current system

The system that we propose in this research is not a finished product; it is instead a foundation for SSC-ICT and other organizations that make use of ticket management systems to extract useful information from their ticket data. Not all components are therefore optimized. However, the system in its current stage already has multiple uses. We now describe these usages of the different components.

Categorization

We built a categorization methodology using LSI to identify categories in the ticket data and cluster them accordingly. Our results show that over 210.000 tickets, it manages to cluster 88% of them in one of 117 clusters (see Appendix F for an overview of the clusters and number of tickets clustered accordingly). The categorization is problem-focused rather than organization focused which is the current categorization of the TopDesk system; it is therefore of added value to the system. This categorization can be used for simple data analysis request which we encountered during our research period like: "how many tickets are about Blackberry in 2018" or "How many status inquiries (status navraag) have there been inquired in the last month?". These are Busines Intelligence requests.

Furthermore, the categories are easily matched to the timestamps which are part of the ticket data in order to provide high-level anomaly detection, due to that in our system the ticket-ids always remain connected to the processed text. Thresholds for the number of incoming tickets over a specific period for specific categories could be set, and on trespassing, a pop-up or message could be triggered. If this, for instance, is matched to the Printer category, an outfall of the Xerox printing process is quickly identified.

Intent identification

Aside from the use of this component for the system the results of this process have more uses. Namely, FAQ extraction, Business Intelligence, and Anomaly Detection. During the research process, the results from POS Patterns were used for a project in which a nationwide Frequently Askes Question-list (FAQ) for the SSC-ICT website was created. The project members did not have any knowledge of the most occurring problems; their guess was that password reset is an accurate one, for which they were right. The results of the intent identification component, may it not be optimized yet, provided them with insights on a data perspective on the most occurring problems. We provided them with 450 intents occurring more than 20 times in the last year.

The Front-end application

The front-end application is meant to be used by the customers of SSC-ICT. However, we recommend first testing and improving the system further in a test-environment. This service desk call-center is a good environment for this, and the application would be useful for them as well. Especially for new operators that do not know the main problems and solutions about the domain, we think this system is very useful. We believe that when they know that the system learns from the feedback that they provide, they will be motivated to do so as well.

7.2 System improvements

In this section, we describe future improvements that would improve the system. These are: Label generation, GuidedLDA for reinforcement learning, Golden set creation for topic count determination, Root Cause Analysis, and reinforcement learning.

We believe that by applying these improvements by a team of one or two programmers the system's performance can be improved by up to 50% within half a year of programming.

Synonyms

A method to improve LDA-clustering is by applying synonyms. These can be applied in the way we did with the POS Patterns, by replacing the input terms with their alpha synonyms. LDA is known to identify synonymous structures itself, but in the case of some categories, this is not possible due to their small size.

Stop words

For stopwords, we used a general Dutch language stopword list. However, we think the system's results can be easily improved by adding domain synonyms as well. Examples that we saw in the clustering process are Dutch versions of the words colleague, madam/sir that would get their cluster. These are easily identifiable stop words that will always be relevant.

Label generation

A disadvantage of Topic Modelling to POS tagging is that the labels of LDA are very unclear; they are merely a summation of keywords that are used to from the topic. However, there are label generation techniques available that create a summarizing label for a collection of documents. In this case, these documents are then the tickets that are part of the intent's cluster.

GuidedLDA for reinforcement learning

GuidedLDA is an adaptation of LDA that is discovered in 2012 by Jagadeesh Jagarlamudi et al. (2012) and made public in a Python library in 2017. The concept is that where LDA is entirely unsupervised, there is no way to influence the topics apart from the topic count, GuidedLDA is. Using "seeds" certain words can be given priority for specific topics with a weight for the height of the priority. We have attempted it for the categorization components, and even though it did not work very well for that, we are pretty sure it does work for intent identification, for the same reason as for why LDA works for intent identification and not for categorization.

In combination with reinforcement learning, individual clusters can be prioritized or fixed by creating a seed for them. By for instance making users able to classify intents as correct or incorrect, reversed keyword identification can identify the seeds which are then added to the GuidedLDA script's resources. From a programmer's point of view, GuidedLDA only extra requires a list of seeds to provide, which makes it very intuitive.

Golden set creation for topic count determination

In this research, we applied the often-used coherence value for finding the optimal topic count. However, due to that this is an internal clustering evaluation methodology, this has its limits which we also encountered. Another way of determining the optimal topic count that we suggest is that of optimizing our proposed evaluation score for each of the categories. For this, a small set of tickets of the category needs to be clustered the way we clustered the evaluation sets, this does take some time, but even from manually clustering about 100 tickets we think it is useful, as the model is trained on the complete corpus. There is a risk of overfitting, so the more tickets clustered, the better, but on the other hand, we believe using even small sets is more accurate and trustworthy than using no method, determining the topic count manually.

Root cause analysis

In this research, we chose not to apply root cause analysis as it was not a priority and we believed at the time that we had too little information for this. However, at this moment we believe it does have use and may be incorporated in the future. Root Cause Analysis in QA-pair generation is the process of looking at the cause of an intent in order to better classify it. Like mentioned in S. Agarwal et al. (2017)the cause can also be deducted from the action that is applied on the ticket, written in the action field in SSC-ICT's ticket set. The intuition behind this is that similar problems also have similar actions. Thus, by analyzing the action fields of an intent cluster, and compare it to that of other clusters, one can potentially merge two clusters that were initially identified by the system as separate but in reality, are not. A step further is to identify synonyms from this process.

Reinforcement learning

A simple, intuitive way to improve the system using reinforcement learning is by pointing feedback back to terms. For instance, when a feedback mechanism points to an action cluster being not accurate, one could combine all these clusters and build a classifier that can classify actions as useful or not. FastText, a technology created by Facebook in 2016, can classify short texts using neural networks very accurately.

Applications

These improvements improve the accuracy of the intent identification process. Label generation makes way for more intuitive results that can be provided to the customer. A potential application would then be a knowledge base for public use. The current knowledge base version contains many faults thus is not yet operable for public use.

8 Discussion

In this research, we design a QA-pair generation system and a prototype service-desk knowledge-base application as front-end. Part of this research are some unique experiments, designs, and findings. We applied categorization methodology before applying the intent identification process; we showed that in combination with this categorization, LDA works best for intent identification which has not been shown before in this research field in a practical setting. In order to evaluate the results of both the POS Patterns and that of LDA, we used a unique combination of evaluation measures of which one we designed ourselves. We designed an external evaluation methodology which does not require a clustering structure on beforehand and is unique in the research field and arguably better than all other options due to its logic.

Furthermore, we showed how Word2vec could be used for synonym detection and showed the improvement of the results of these synonyms compared to before applying them. Furthermore, the system that we designed is very easily applicable to new datasets; it requires little manual labeling. We now describe each of these topics in more detail.

Categorization

Because of the very high variety of the SSC-ICT dataset we were bound to find a method to reduce the variation of the ticket dataset. Our solution is splitting up the corpus automatically using a single-term LSI-based methodology, after which multiple, low variety corpora, categories, can be clustered independently. We posit that this decision is what made it possible for LDA to be applied successfully. This solution has not been used in any of the research that we reviewed, and this might very well be the reason why they skipped LDA since we also got useless results when attempting LDA on the corpus without categorizational clustering.

LDA vs. POS Patterns

Jan, Chen, and Ide (2014) is the only article of our related works to mention LDA for intent identification in incident tickets. Our research confirms this. A downside of POS patterns to LDA is the case when no verbs are found in the ticket description. In our dataset, this was a big problem, with a coverage of less than 40% for the POS patterns. LDA does not look at the syntactical meaning of terms but rather at relational meaning. In a high variety corpus this is very difficult, but due to our high-level categorization, this was not an issue. Regarding the potential of LDA to POS patterns, we believe LDA surpasses the latter by miles. With more and more feedback, more advanced topics can be identified, and in combination with GuidedLDA, stored as well.

Custom Evaluation score

The benefit but at the same time also the problem of working with POS patterns or LDA is that the results have no predefined structure, or labels, on which they can be evaluated. This probably explains why none of the related works provide a robust evaluation methodology for these techniques, at least not one that is not external, because internal evaluation is not suitable for intent identification due to the high complexity in the meaning of the intents. Plus, the fact that internal evaluation methodologies use the same features that the clustering methodologies do, which is why they are very prone to overfitting. Our evaluation methodology computes the proportion of mutual tickets for each ticket in its parent cluster, compared to a golden test set. The logic is complete. We showed minimal quality levels using the two extreme situations that are known to cause for high evaluation scores: all items in unique clusters and all items in the same cluster. Moreover, we proved that both the POS patterns and

LDA scored better on these than they did. A point of interest is that the scores for the evaluations are relatively low. This is an accurate point, but there are many reasons why they could be so low. One that we are sure of happens is when a very large golden set cluster is in the system's version split into two still relatively large clusters, which halves the evaluation scores of these tickets which is of significant influence on the overall evaluation score. Improvements of the intent identification process, especially cluster merging using Root Cause Analysis or Reinforcement Learning could easily avoid this problem and thus have a significant impact of improving the system on the evaluation score and in actual practice as well.

Word2vec

We showed the potential of Word2vec in the field of synonym detection. Even though the increase in evaluation score was minimal, the technique did work in identifying over 300 synonyms. We think Word2vec is especially useful in a system that is applied to many different datasets due to the speed with which it generates synonyms (once the scripts are built, because figuring that out may take some time). The resulting synonym sets do however require manual correction because in some cases Word2vec may find words that are similar, but rather than synonymously similar, similar in for instance a hierarchically dependent way. However, checking synonym sets is a very intuitive process and takes very little time. We checked 100 synonym sets in less than 10 minutes, which is much and much faster than identifying synonyms manually.

Dynamic/scalable system

Based on the system characteristics that we identified in chapter 2, we tried to minimize the manual required effort in every way possible. The result is a system that we can apply to any new, structurally similar dataset (short descriptions + action fields) and provide a working system in less than a day. This is not only useful for SSC-ICT, who are adding a new large ministry in their TopDesk system soon: the Ministry for External Affairs, but also for the company TopDesk itself. Topdesk has hundreds of large companies as customers but does not have anything related to this topic. After consultation with the public-sector business director and one of the 5 data scientists of TopDesk, it is confirmed that they do not have the resources for starting such a design project, even though they did find it very interesting. Potential future research to come?

9 Conclusion

In this chapter, we answer the research question and the main research question that we posed in chapter

What components, techniques, and characteristics of QA pair generation systems from related works?

From a literature review, we identified the components Intent Identification, Root Cause Analysis, Action Recommendation, and Reinforcement Learning. We added to this the component of Categorization due to the large dataset and high variety of tickets of SSC-ICT. We also identified techniques from the literature review. We grouped them in the groups Pre-processing, Clustering, Synonyms, and Reinforcement Learning. For categorization, we identified LSI, LDA and POS patterns. We identified the characteristics by looking at the differences between the related works and our research case. The characteristics are Language, Size of the dataset, Length of documents, Variation in intents, Variation in domains, The speed of structural change in topics, Amount of future development, Amount of manual work availability, Number of potential users, Privacy restrictions.

What potentially useful, other techniques are there?

In order to answer this question, we had contact with multiple data science companies and shifted through online fora and other documentation. Topic Modelling was a big topic that we encountered in many different areas. Even though it was most often used to find general topics in large documents, we found we had to give it a try, especially with the evaluation results of Jan, Chen, and Ide (2014).

Word2vec is a well known and high-quality method for doing all sorts of things with word relationships and showed good promise for synonym detection.

In (Vlasov et al., 2017) Bi-grams were manually applied in order to replace specific multi-noun keywords. When we encountered the deep learning bi-gram detection possibilities of the Gensim library, we knew we had to give it a try.

What are the characteristics of the SSC-ICT dataset?

, In chapter 3 we described the ticket data in much detail. We explained the eight most relevant fields of the ticket data and how we used those fields to choose a suitable dataset. Furthermore, we decided that for the intent identification and result recommendation we would focus on respectively the short description and the action field. The request field was too noisy, too long and too inconsistent to put effort into. For the categorization fields, we had analyzed the contents of the tickets and found that 33% of the tickets were manually categorized wrongly. Furthermore, we did not find the subcategories very specific, and their coverage was too inconsistent as well: 50% of the tickets was categorized among three subcategories.

How can QA pair quality best be measured?

For answering this research question we consulted some literature reviews, the general consensus scientific research was that there are some types of evaluation for NLP systems and some guidelines, but that overall it often is unique for the dataset and the context.

Due to that, the system consists of multiple components that all have their own input, we decided that we needed component-based evaluation methods rather than only end-result evaluation. For the categorization component, we chose for a score for the number of tickets that were categorized as well as the number of categories that would result from the component. Furthermore, we set some boundary conditions to which the categories needed to comply: Unique, Value adding and Hierarchically independent.

Regarding the intent identification, we learned from the literature review that there are two types of evaluation: internal and external. We decided that we required external evaluation for our research, though generally, these methods required a predefined structure or accurate labels, things we both did not have. Therefore we devised our own evaluation technique to measure the quality of the structures. One that does not require labels or structure and uses a golden ticket set to score results. In order to avoid the risk of overfitting, we created three golden cluster sets of three different sized and also different type of categories. Furthermore, we determined that the number of tickets covered, along with a threshold for intent-size was relevant for evaluation, as well as processing speed.

For the action recommendation component, we decided that the percentage of unique and useful actions proposed is a good measure. However, this is meant for future use of the system, thus not evaluated in this research, in contrary to the other two components. The reinforcement learning component also requires feedback to be able to be evaluated. Furthermore, its results can be seen in increased results for the other three components rather than having its own measure.

What is the minimal quality level needed to produce relevant performance measures?

We determined the minimal quality level from a customer satisfaction point of view. The system should perform at a level in which it improves the customer satisfaction. The system should therefore lead to a successful answer often enough to be used by a good amount of people.

How can QA pairs best be used at SSC-ICT?

In chapter 7 we describe the way the system of this research can be used at SSC-ICT. Furthermore, we describe the improvements that can be made to the system in order to increase the QA-pair quality. The results of the proposed system without improvement can be used for business intelligence, FAQ creation, and interactive knowledge base. Especially the high-level categories are a trustworthy result from the system that can be directly used for business insights that are not possible as of yet. The QA-pairs are as of yet less trustworthy but are useful for internal use by for instance new operators with no knowledge of the domain, and for FAQ creation with manual correction. The knowledge base function provides feedback for the reinforcement learning system that is used to improve the system, so we highly recommend implementing this feature as well.

10 Future research

In this chapter, we describe the potential future research that this research implies.

The main subjects that this research puts forward which are not extensively researched are that of the use of Topic Modelling (LDA) and reinforcement learning for improving the intent identification component.

LDA is generally used for identifying general topics from large documents and is the single most used algorithm for this subject. However, Jan et al.(2014) and this research show that LDA can also be used for identifying unique intents in low variety datasets. The downside of Topic modeling has always been that it is completely unsupervised and that apart from determining the amount of topics there is no way to influence this process. However, as of 2017, GuidedLDA has been discovered, a method to seed keywords in LDA topics, steering the algorithm in a preferred direction to identify topics around. GuidedLDA has however barely been researched yet. We are curious to see how far this steering can go. Its potential seems unlimited, reaching towards topic databases in which topics instead of lexical keywords are stored, with hundreds of weighted terms per topic.

Reinforcement learning is due to its feedback requirements also very little described in literature. However, the same for this subject counts that it provides great potential for companies like SSC-ICT that cover large amounts of users. Companies like Google are highly invested in this subject but keep their techniques a secret. It would be interesting to see more information come available to what and how human feedback is applied in order to improve text clustering.

11 References

Abraham, D. M., Spangler, W. E., & May, J. H. (1991). Expertech: Issues in the design and development of an intelligent help desk system. *Expert Systems With Applications*, *2*(4), 305–319. http://doi.org/10.1016/0957-4174(91)90037-F

Acorn, T. L. (1992). SMART: Support Management Automated Reasoning Technology for Compaq Customer Service. *IAAI-92 Proceedings*.

Agarwal, S., Aggarwal, V., Akula, A. R., Dasgupta, G. B., & Sridhara, G. (2017). Automatic problem extraction and analysis from unstructured text in IT tickets. *IBM Journal of Research and Development*. http://doi.org/10.1147/JRD.2016.2629318

Agarwal, S., Sindhgatta, R., & Sengupta, B. (2012). SmartDispatch: enabling efficient ticket dispatch in an IT service environment. *Proceedings of the 18th ACM …*, 1393–1401. http://doi.org/10.1145/2339530.2339744

Blaz, C. C. A., & Becker, K. (2016). Sentiment analysis in tickets for IT support. In *Proceedings of the 13th International Workshop on Mining Software Repositories - MSR '16* (pp. 235–246). http://doi.org/10.1145/2901739.2901781

Bozdogan, C., & Zincir-Heywood, N. (2012). Data mining for supporting IT management. In *Proceedings of the 2012 IEEE Network Operations and Management Symposium, NOMS 2012* (pp. 1378–1385). http://doi.org/10.1109/NOMS.2012.6212079

Chan, C. W., Chen, L. L., & Geng, L. (2000). Knowledge engineering for an intelligent case-based system for help desk operations. *Expert Systems with Applications*, *18*(2), 125–132. http://doi.org/10.1016/S0957-4174(99)00058-5

Chang, K. H., Raman, P., Carlisle, W. H., & Cross, J. H. (1996). A self-improving helpdesk service system using case-based reasoning techniques. *Computers in Industry*, *30*(2), 113–125. http://doi.org/10.1016/0166-3615(96)00033-4

Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., & Wirth, R. (2000). Crisp-Dm 1.0. *CRISP-DM Consortium*, 76. http://doi.org/10.1109/ICETET.2008.239

Cheung, C. F., Lee, W. B., Wang, W. M., Chu, K. F., & To, S. (2003). A multi-perspective knowledge-based system for customer service management. *Expert Systems with Applications*. http://doi.org/10.1016/S0957-4174(02)00193-8

Choe, P., Lehto, M. R., Shin, G. C., & Choi, K. Y. (2013). Semiautomated identification and classification of customer complaints. *Human Factors and Ergonomics In Manufacturing*, *23*(2), 149–162. http://doi.org/10.1002/hfm.20325

Datanyze. (2019). TOPdesk Market Share in Netherlands and Competitor Report | Compare to ServiceNow, Freshservice, Jira Service Desk | Datanyze. Retrieved March 19, 2019, from https://www.datanyze.com/market-share/itsm/Netherlands/topdesk-market-share

Davenport, T. H., & Klahr, P. (1998). *Managing Customer Support Knowledge*. California Mangement Review, Vol. 40, No. 3

Dhoolia, P., Chugh, P., Costa, P., & Gantayat, N. (2017). A cognitive system for business and technical support : A case study, *61*(1), 74–85. http://doi.org/10.1147/JRD.2016.2631398

Dhoolia, P., Chugh, P., Costa, P., Gantayat, N., Gupta, M., Kambhatla, N., … Saxena, M. (2017). A cognitive system for business and technical support: A case study. http://doi.org/10.1147/JRD.2016.2631398

El Sawy, O. A., & Bowles, G. (1997). Redesigning the Customer Support Process for the Electronic Economy: Insights from Storage Dimensions. *MISQ*, *21*(4), 457. http://doi.org/10.2307/249723

Foo, S., S.C, H., Leong, P. C., & Liu, S. (2000). An Iintegrated Help Desk Support for Customer Services over the World Wide Web - A Case Study. *Computers in Industry*, *41*(2), 129–145.

García-Pardo, J. Á., Barberá, S. H., Ramos-Garijo, R., Palomares, A., Julián, V., Rebollo, M., & Botti, V. (2006). CBR-TM: A new case-based reasoning system for help-desk environments. *Frontiers in Artificial Intelligence and Applications*, *141*(January 2006), 833–834. Retrieved from http://www.scopus.com/inward/record.url?eid=2-s2.0-84885995660&partnerID=tZOtx3y1

Göker, M., & Roth-Berghofer, T. (1999). Development and utilization of a case-based help-desk support system in a corporate environment. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (Vol. 1650, pp. 132–146). http://doi.org/10.1007/3-540-48508-2_10

González, L. M., Giachetti, R. E., & Ramirez, G. (2005). Knowledge management-centric help desk: Specification and performance evaluation. *Decision Support Systems*, *40*(2), 389–405. http://doi.org/10.1016/j.dss.2004.04.013

Gupta, R., Prasad, K. H., & Mohania, M. (2008). Automating ITSM incident management process. In *5th International Conference on Autonomic Computing, ICAC 2008* (Vol. 1, pp. 141–150). http://doi.org/10.1109/ICAC.2008.22

Heras, S., García-pardo, J. Á., Ramos-garijo, R., Palomares, A., Botti, V., Rebollo, M., & Julián, V. (2009). Multi-domain case-based module for customer support. *Expert Systems With Applications*, *36*(3), 6866–6873. http://doi.org/10.1016/j.eswa.2008.08.003

Ho Kang, B., Yoshida, K., & Compton, P. (1997). Help desk system with intelligent interface. *Applied Artificial Intelligence*, *11*, 611–631.

Iwai, K., Iida, K., Akiyoshi, M., & Komoda, N. (2010). A help desk support system with filtering and reusing e-mails. In *IEEE International Conference on Industrial Informatics (INDIN)* (pp. 321–325). http://doi.org/10.1109/INDIN.2010.5549401

Jan, E., Chen, K., & Ide, T. (2014). A Probabilistic Concept Annotation for IT Service Desk Tickets. In *Proceedings of the 7th International Workshop on Exploiting Semantic Annotations in Information Retrieval - ESAIR '14* (pp. 21–23). http://doi.org/10.1145/2663712.2666193

Jordán, J., Heras, S., & Julián, V. (2011). A customer support application using argumentation in Multi-Agent Systems. In *Fusion 2011 - 14th International Conference on Information Fusion* (pp. 772–778).

Kang, Y., & Zaslavsky, A. (2010). A knowledge-rich similarity measure for improving IT incident resolution process. *Proceedings of the 2010 …*, 1781–1788. http://doi.org/10.1145/1774088.1774466

Kim, H., & Seo, J. (2008). Cluster-based FAQ retrieval using latent term weights. *IEEE Intelligent Systems*, *23*(2), 58–65. http://doi.org/10.1109/MIS.2008.23

Kiyota, Y., Kurohashi, S., & Kido, F. (2003). Dialog Navigator: A Question Answering System based on Large Text Knowledge Base. *Journal of Natural Language Processing*, *10*(4), 145–175. http://doi.org/10.5715/jnlp.10.4_145

Kongthon, A., Sangkeettrakarn, C., Kongyoung, S., & Haruechaiyasak, C. (2009). Implementing an online help desk system based on conversational agent. In *Proceedings of the International Conference on Management of Emergent Digital EcoSystems - MEDES '09* (p. 450). http://doi.org/10.1145/1643823.1643908

Kozakov, L., Park, Y., Fin, T., Drissi, Y., Doganata, Y., & Cofino, T. (2004). Glossary extraction and utilization in the information search and delivery system for IBM Technical Support. *IBM Systems Journal*, *43*(3), 546–563. http://doi.org/10.1007/3-540-32394-5_20

Li, H., & Zhan, Z. (2012). Machine learning methodology for enhancing automated process in IT incident management. In *Proceedings - IEEE 11th International Symposium on Network Computing and Applications, NCA 2012* (pp. 191–194). http://doi.org/10.1109/NCA.2012.28

Liu, Y., Li, Z., Xiong, H., Gao, X., & Wu, J. (2010). Understanding of Internal Clustering Validation Measures. IEEE International Conference on Data Mining Understanding http://doi.org/10.1109/ICDM.2010.35

Mani, S., Sankaranarayanan, K., Sinha, V. S., & Devanbu, P. (2014). Panning requirement nuggets in stream of software maintenance tickets. In *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering - FSE 2014* (pp. 678–688). http://doi.org/10.1145/2635868.2635897

Marcu, P., Grabarnik, G., Luan, L., Rosu, D., Shwartz, L., & Ward, C. (2009). Towards an optimized model of incident ticket correlation. In *2009 IFIP/IEEE International Symposium on Integrated Network Management, IM 2009* (pp. 569–576). http://doi.org/10.1109/INM.2009.5188863

Miao, G., Moser, L. E., Yan, X., Tao, S., Chen, Y., & Anerousis, N. (2010). Generative models for ticket resolution in expert networks. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '10* (p. 733). http://doi.org/10.1145/1835804.1835897

Motahari-Nezhad, H. R., & Bartolini, C. (2011). *Next Best Step and Expert Recommendation for Collaborative Processes in IT Service Management* (Vol. 6896). http://doi.org/10.1007/978-3-642-23059-2

Motahari Nezhad, H. R., Bartolini, C., & Joshi, P. (2011). Analytics for similarity matching of IT cases with collaboratively-defined activity flows. In *Proceedings - International Conference on Data Engineering* (pp. 273–278). http://doi.org/10.1109/ICDEW.2011.5767639

Osinski, S., Stefanowski, J., & Weiss, D. (2004). Lingo: Search Results Clustering Algorithm Based on Singular Value Decomposition. *Advances in Soft Computing, Intelligent Information Processing and Web Mining, Proceedings of the International IIS: IIPWM '04 Conference*, 359–368. http://doi.org/10.1007/978-3-540-39985-8_37

Palshikar, G. K., Vin, H. M., Mudassar, M., & Natu, M. (2010). Domain-driven data mining for IT infrastructure support. In *Proceedings - IEEE International Conference on Data Mining, ICDM* (pp. 959–966). http://doi.org/10.1109/ICDMW.2010.132

Paroubek, P., Chaudiron, S., Hirschman, L., Paroubek, P., Chaudiron, S., & Hirschman, L. (2010). Principles of Evaluation in Natural Language Processing To cite this version : HAL Id : hal-00502700 Principles of Evaluation in Natural Language Processing, *48*(1), 7–31.

Potharaju, R., Chan, J., Hu, L., Nita-rotaru, C., Wang, M., Zhang, L., & Jain, N. (2015). ConfSeer: Leveraging Customer Support Knowledge Bases for Automated Misconfiguration Detection. *Proceedings of the 41st International Conference on Very Large Data Bases*, 1828–1839. http://doi.org/10.14778/2824032.2824079

Potharaju, R., & Nita-rotaru, C. (2013). Juggling the Jigsaw: Towards Automated Problem Inference from Network Trouble Tickets. *Nsdi*, 127–141.

Rahman, I., Alarifi, A., Eden, R., & Sedera, D. (2014). Archival analysis of service desk research: New perspectives on design and delivery. *25th Australasian Conference on Information Systems*, 8–10. http://doi.org/10.1177/0741713604268894

Resnik, P., & Lin, J. (2010). Evaluation of NLP Systems. *The Handbook of Computational Linguistics and Natural Language Processing*, 271–295. http://doi.org/10.1002/9781444324044.ch11

Roth-berghofer, T., & Roth-berghofer, T. R. (2004). Learning from HOMER , a case- based help desk support system Learning from HOMER , (June 2014). http://doi.org/10.1007/978-3-540-25983-1

Russell, S., & Norvig, P. (2013). *Artificial Intelligence A Modern Approach. Zhurnal Eksperimental'noi i Teoreticheskoi Fiziki*. http://doi.org/10.1017/S0269888900007724

Samejima, M., & Akiyoshi, M. (2013). *A Help Desk Support System Based on Relationship between Inquiries and Responses* (Vol. 484). http://doi.org/10.1007/978-3-642-37932-1

ServiceNow, & Devoteam. (2018). The AI Revolution. Retrieved from https://www.servicenow.com/lpayr/ai-revolution.html

Shanavas, N., & Asokan, S. (2015). Ontology-Based Document Mining System for IT Support Service. *Procedia - Procedia Computer Science*, *46*(Icict 2014), 329–336. http://doi.org/10.1016/j.procs.2015.02.028

Shao, Q., Chen, Y., Tao, S., Yan, X., & Anerousis, N. (2008). Efficient ticket routing by resolution sequence mining. In *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery* *and data mining - KDD 08* (p. 605). http://doi.org/10.1145/1401890.1401964

Sharma, A. R., & Kaushik, P. (2017). Literature survey of statistical, deep and reinforcement learning in natural language processing. *Proceeding - IEEE International Conference on Computing, Communication and Automation, ICCCA 2017*, *2017–Janua*, 350–354. http://doi.org/10.1109/CCAA.2017.8229841

Sneiders, E. (2009). Automated FAQ Answering with Question-Specific Knowledge Representation for Web Self-Service. *Proceedings of the 2nd International Conference on Human System Interaction (HSI'09),* 298–305.

Sun, P., Tao, S., Yan, X., Anerousis, N., & Chen, Y. (2010). Content-Aware Resolution Sequence Mining for Ticket Routing. IBM T. J. Watson Research Center

Takano, A., Yurugi, Y., & Kanaegami, A. (2000). Procedure based help desk system. *Proceedings of the 5th International Conference on Intelligent User Interfaces - IUI '00.* http://doi.org/10.1145/325737.325868

Talamo, M., Povilionis, A., Arcieri, F., & Schunck, C. H. (2016). Providing Online Operational Support for Distributed , Security Sensitive Electronic Business Processes, 49–54.

Thurman, D. a., Tracy, J. S., & Mitchell, C. M. (1997). Design of an intelligent Web-based help desk system. *1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation*, *3.* http://doi.org/10.1109/ICSMC.1997.635192

Vehviläinen, A., Hyvönen, E., & Alm, O. (2006). A semi-automatic semantic annotation and authoring tool for a library help desk service. In *CEUR Workshop Proceedings* (Vol. 209). http://doi.org/10.4018/978-1-59904-877-2.ch007

Vlasov, V., Chebotareva, V., Rakhimov, M., & Kruglikov, S. (2017). AI User Support System for SAP ERP. Journal of Physics: Conference Series 913

12 Appendix

12.1 Appendix A: literature review articles

Author	Year	Ontology creation	Pre-processing	Keyword classificatoin	Clustering	Ticket routing	Type of system
(Abraham, Spangler, & May, 1991)	1991	x					Expert system
(Acorn, 1992)	1992						CBR system
(Chang, Raman, Carlisle, & Cross, 1996)	1996						CBR system
(El Sawy & Bowles, 1997)	1997						CBR system
(Ho Kang, Yoshida, & Compton, 1997)	1997						CBR system
(Thurman, Tracy, & Mitchell, 1997)	1997						CBR system
(Davenport & Klahr, 1998)	1998						Systems overview
(Göker & Roth-Berghofer, 1999)	1999						CBR system
(Chan, Chen, & Geng, 2000)	2000						CBR system
(Takano, Yurugi, & Kanaegami, 2000)	2000			x			CBR system
(Foo, S.C, Leong, & Liu, 2000)	2000	x					Expert system
(Kiyota, Kurohashi, & Kido, 2003)	2002	x	x		x		QA system
(Cheung, Lee, Wang, Chu, & To, 2003)	2003	x			x		Knowledge based system
(Roth-berghofer & Roth-berghofer, 2004)	2004						CBR system
(Kozakov et al., 2004)	2004	x					Knowledge base system
(González, Giachetti, & Ramirez, 2005)	2005						Expert system
(García-Pardo et al., 2006)	2006				x		CBR system
(Gupta, Prasad, & Mohania, 2008)	2008	x	x	x			Knowledge base system
(Kim & Seo, 2008)	2008		x	x			QA system
(Vehviläinen, Hyvönen, & Alm, 2006)	2008	x	x	x	x		QA system/CBR
(Shao, Chen, Tao, Yan, & Anerousis, 2008)	2008				x	x	Ticket recommender

(Heras et al., 2009)	2009						CBR system
(Kongthon, Sangkeettrakarn, Kongyoung, & Haruechaiyasak, 2009)	2009	x	x	x			QA system
(Sneiders, 2009)	2009		x		x		QA system
(Marcu et al., 2009)	2009			x	x	x	Ticket recommender
(Kang & Zaslavsky, 2010)	2010	x	x	x	x		CBR system
(Iwai, Iida, Akiyoshi, & Komoda, 2010)	2010	x	x	x	x		QA system: help desk
(Palshikar, Vin, Mudassar, & Natu, 2010)	2010			x		x	Ticket recommender
(Sun, Tao, Yan, Anerousis, & Chen, 2010)	2010		x	x	x	x	Ticket recommender
(Miao et al., 2010)	2010		x	x	x	x	Ticket recommender
(Jordán, Heras, & Julián, 2011)	2011						CBR system
(Motahari-Nezhad & Bartolini, 2011)	2011		x	x		x	Ticket recommender
(Motahari Nezhad, Bartolini, & Joshi, 2011)	2011		x	x		x	Ticket recommender
(Bozdogan & Zincir-Heywood, 2012)	2012	x	x	x	x		Knowledge base system
(Shivali Agarwal, Sindhgatta, & Sengupta, 2012)	2012		x	x	x	x	Ticket recommender
(Li & Zhan, 2012)	2012	x	x	x	x		Ticket recommender
(Choe, Lehto, Shin, & Choi, 2013)	2013	x	X	x	x		Knowledge base system
(Potharaju & Nita-rotaru, 2013)	2013	x	X	x			Knowledge bases system
(Samejima & Akiyoshi, 2013)	2013	x		x	x		QA system
(Rahman, Alarifi, Eden, & Sedera, 2014)	2014						Systems overview
(Jan et al., 2014)	2014	x	X	x			Ticket recommender
(Shanavas & Asokan, 2015)	2015	x					Knowledge base system
(Potharaju et al., 2015)	2015	x	X	x			Knowledge base with automated issue detection system
(Blaz & Becker, 2016)	2016			x	x		Knowledge base system
(Talamo, Povilionis, Arcieri, & Schunck, 2016)	2016						Ticket recommender
(S Agarwal et al., 2017)	2017	x	X	x	x		Knowledge base system
(Dhoolia, Chugh, Costa, & Gantayat, 2017)	2017	x	X	x	x		QA system

(Vlasov et al., 2017)	2017	x		x	x		QA system

12.2 Appendix B: Literature review methodology

To answer the research question a structured literature review was performed. Scopus and Google Scholar were used for scientific libraries in order to search for scientific papers. First, an initial search query was designed in order to find a first selection of relevant articles. This query was: (knowledge OR information OR system) AND (“customer support” OR “user support” OR “technical support” OR “help desk”). A total of 205 articles was found. The articles were scanned on article title and abstract for relevance to the subject. Citation count and year of publishing was taken into account: articles with a low citation count needed to be published relatively recently in order to make it through the selection. This resulted in a set of 62 articles.

Next, these articles were read fully in order to be more selective about the relevance, the result of this was 27 articles. During this step a new keyword “ticket” was identified and a couple of relevant articles were added. Forward and backward snowballing technique was applied on the resulting set in order to find more articles. This process was repeated at least three times until no new articles were found. This resulted in a set of 49 articles.

These articles were then coded and sorted in categories for each of the knowledge processes, system type, and other information relevant to be able to quickly look up an overview of the articles. A partial overview is shown in table 1.

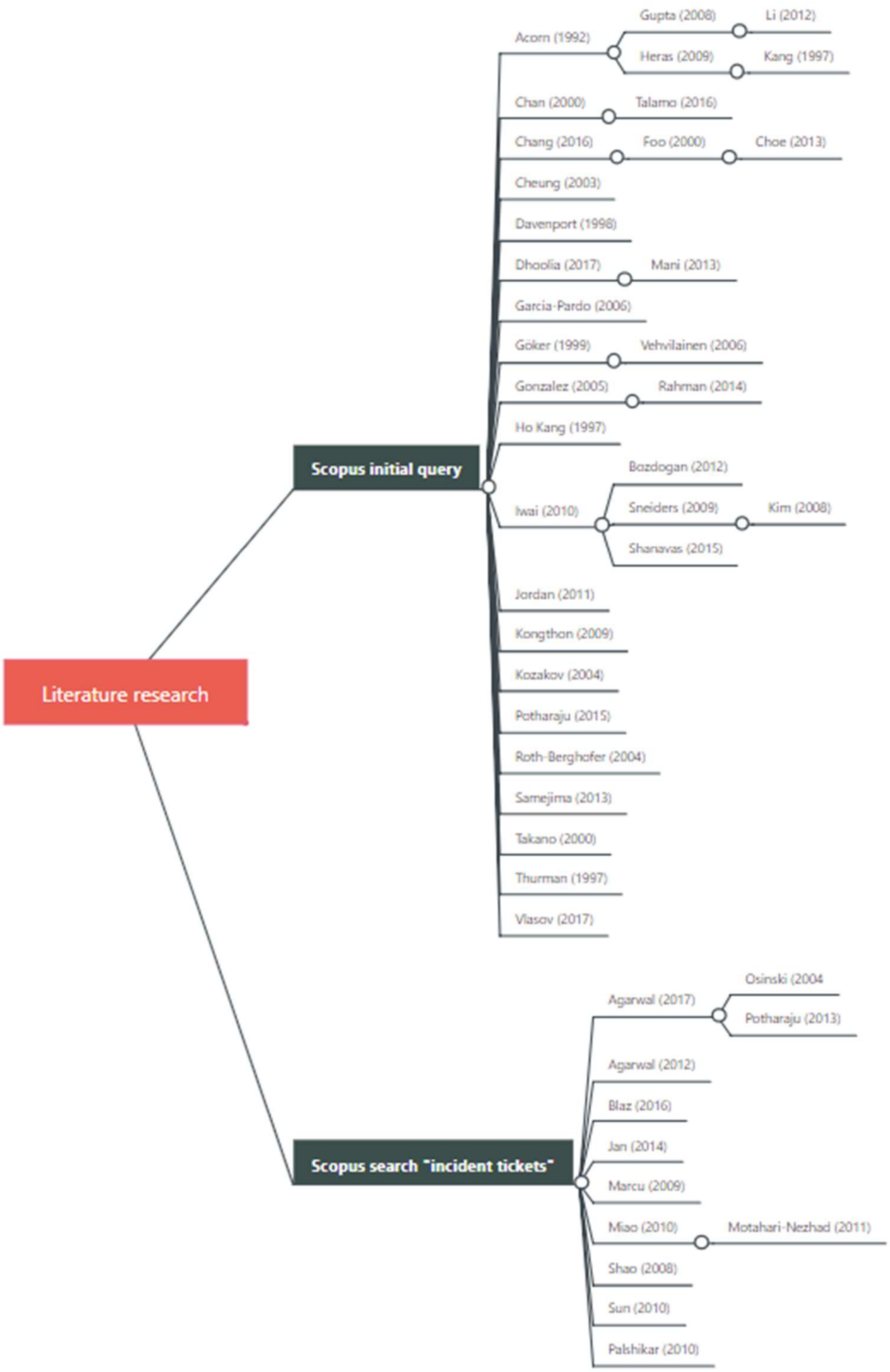
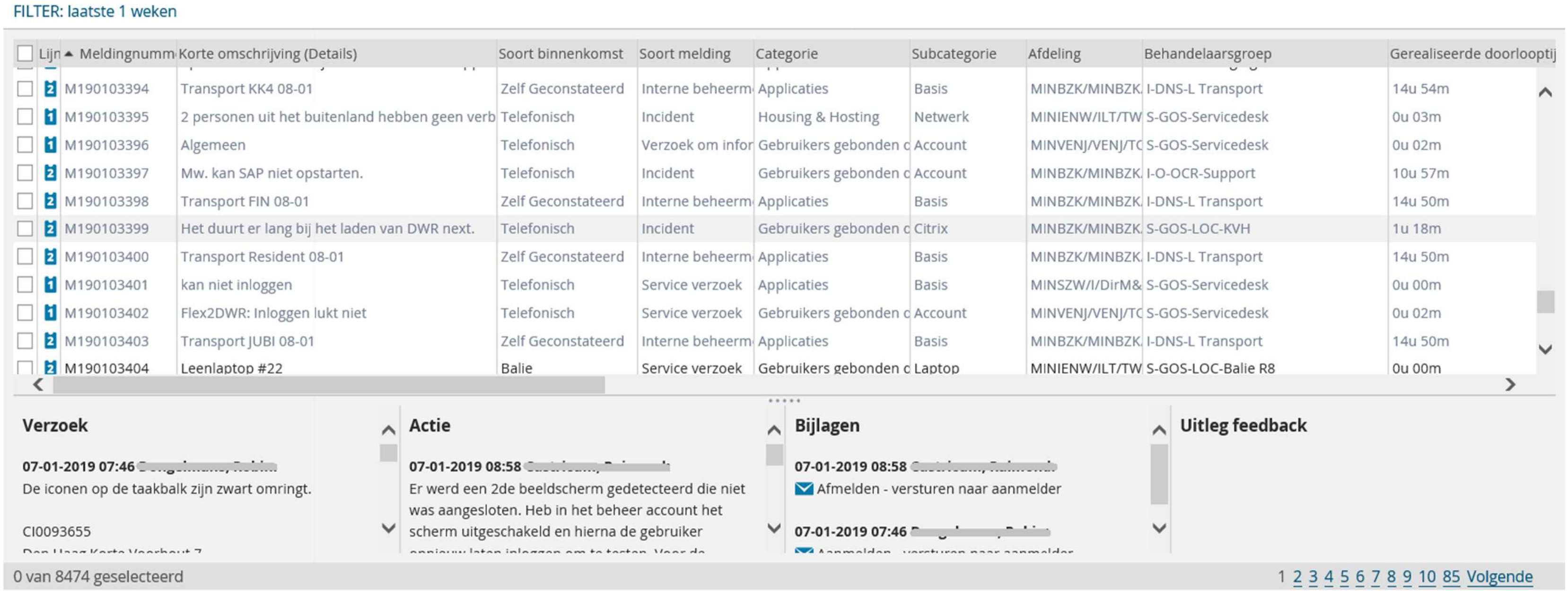


Figure 19: Literature review process visualized

12.3 Appendix C: Ticket overview in TopDesk System



FILTER: laatste 1 weken

□ Lijn ▲	Meldingnumm	Korte omschrijving (Details)	Soort binnenkomst	Soort melding	Categorie	Subcategorie	Afdeling	Behandelaarsgroep	Gerealiseerde doorlooptij
□ 2	M190103394	Transport KK4 08-01	Zelf Geconstateerd	Interne beheerm	Applicaties	Basis	MINBZK/MINBZK	I-DNS-L Transport	14u 54m
□ 1	M190103395	2 personen uit het buitenland hebben geen verb	Telefonisch	Incident	Housing & Hosting	Netwerk	MINIENW/ILT/TW	S-GOS-Servicedesk	0u 03m
□ 1	M190103396	Algemeen	Telefonisch	Verzoek om infor	Gebruikers gebonden c	Account	MINVENJ/VENJ/TC	S-GOS-Servicedesk	0u 02m
□ 2	M190103397	Mw. kan SAP niet opstarten.	Telefonisch	Incident	Gebruikers gebonden c	Account	MINBZK/MINBZK	I-O-OCR-Support	10u 57m
□ 2	M190103398	Transport FIN 08-01	Zelf Geconstateerd	Interne beheerm	Applicaties	Basis	MINBZK/MINBZK	I-DNS-L Transport	14u 50m
□ 2	M190103399	Het duurt er lang bij het laden van DWR next.	Telefonisch	Incident	Gebruikers gebonden c	Citrix	MINBZK/MINBZK	S-GOS-LOC-KVH	1u 18m
□ 2	M190103400	Transport Resident 08-01	Zelf Geconstateerd	Interne beheerm	Applicaties	Basis	MINBZK/MINBZK	I-DNS-L Transport	14u 50m
□ 1	M190103401	kan niet inloggen	Telefonisch	Service verzoek	Applicaties	Basis	MINSZW/I/DirM&	S-GOS-Servicedesk	0u 00m
□ 1	M190103402	Flex2DWR: Inloggen lukt niet	Telefonisch	Service verzoek	Gebruikers gebonden c	Account	MINVENJ/VENJ/TC	S-GOS-Servicedesk	0u 02m
□ 2	M190103403	Transport JUBI 08-01	Zelf Geconstateerd	Interne beheerm	Applicaties	Basis	MINBZK/MINBZK	I-DNS-L Transport	14u 50m
□ 2	M190103404	Leenlaptop #22	Balie	Service verzoek	Gebruikers gebonden c	Laptop	MINIENW/ILT/TW	S-GOS-LOC-Balie R8	0u 00m

Verzoek

07-01-2019 07:46 [redacted]
De iconen op de taakbalk zijn zwart omringt.

CI0093655
Den Haag Korte Voorhout 7

Actie

07-01-2019 08:58 [redacted]
Er werd een 2de beeldscherm gedetecteerd die niet was aangesloten. Heb in het beheer account het scherm uitgeschakeld en hierna de gebruiker opnieuw laten inloggen om te testen. Voor de

Bijlagen

07-01-2019 08:58 [redacted]
✉ Afmelden - versturen naar aanmelder

07-01-2019 07:46 [redacted]
✉ Aanmelden - versturen naar aanmelder

Uitleg feedback

0 van 8474 geselecteerd

1 2 3 4 5 6 7 8 9 10 85 Volgende

12.4 Appendix D: List of synonyms

12.4.1 Entity synonyms

website site url

hprm digidoc

inet i-net

pincode pin

monitoring agent

acceptatieomgeving testomgeving

postvak inbox

benaderbaar toegankelijk

oplader oplaadkabel lader voeding adapter usb-c

access acces

adobe acrobat

samenwerkingsruimte samenwerkruimte swr

traag langzaam

token softtoken hardtoken softoken

work works

raac zorro notis

synchroniseert sync synct synchroniseerd

blackberry bb good bbwork goodwork blackberrywork uemclient

simkaart umts sim sim-kaart umts-kaart

wifi wi-fi govroam internetverbinding

pst-map gegevensbestand

aanmeldserver aanmeldingsserver

kamer vergaderzaal zaal

etage verdieping

cloudbook macbook

installatie activatie herinstallatie

ontgrendelcode activatiecode toegangscode pukcode puk ontgrendelingscode activeringscode unlockcode ontgrendelingssleutel installatiecode

toner afvalcontainer container tonerafvalcontainer cassette afdrukmodule

res one

workspace ivanti

mail mails email e-mail

dwr citrix dwr64 dwr-64

win7 w7

win10 w10

vgw vgw-rvb

servicedesk helpdesk

adminsitratie adminstratie

firefox ff

wifi govroam wi-fi

defect kapot

beeldscherm scherm monitor beeld

netwerkverbinding internetverbinding dataverbinding

laptop chromebook

pc computer

schijf g-schijf h-schijf netwerkschijf o-schijf

raar vreemd

usb stick sticks

vergroten uitbreiden

gebruiker klant gebr aanmelder

dekking buitenlanddekking werelddekking

onjuist ongeldig

autoriseren machtigen

postbus mailbox dienstpostbus

uem eum

update upgrade overgang

migratie verhuizing

uitgeleend leen uitleen

afhalen afboeken

proxy proxyserver proxy-server

synergy globe

ongeluk abuis

verbinding connectie

telefoon iphone ipad toestel mobiel samsung smartphone

gebruikersnaam inlognaam

afdeling directie

factuur inkooporder io

mfp xerox

mailadres e-mailadres emailadres

proxymelding proxy-melding

balie servicebalie

diverse meerdere allerlei

machtigingen machtiging

followme followme1

installatie activatie herinstallatie activatiemail heractivatie

bestand document

crasht crashed

gemigreerd overgezet

pagina webpagina

kabel netwerkkabel

replicator portreplicator dockingstation

mappen map submap

wachtwoord ww password

beveiligingsmelding popup pop-up

invoegtoepassing plugin

opdracht printopdracht

geheugen schijfruimte

aub svp

laptops pc's

enorm extreem ontzettend

installeren configureren

mozilla frontmotion

ip mac

weergave layout

virusscanner mcafee

kopieren verslepen

probleem euvel

netwerkschijven schijven

code sleutel

simwissel wissel

database databases

bes12 bes

batterij accu

12.4.2 *Verb synonyms*

vergrendelen locken deactiveren
deblokkeren unlocken heractiveren
synchroniseren synct synchroniseeren
failed hossen alert dwrt certificate
omruilen omwisselen inleveren
benaderen bereiken
verplaatsen slepen verslepen
terugzetten terugplaatsen
weergeven tonen
afvoeren verhuizen
uitgeven uitleveren meegeven
inleveren omruilen omwisselen
aankomen binnenkomen
herstarten rebooten
registreren registeren

printen afdrukken uitprinten
knipperen flikkeren
controleren nakijken
bewaren terugkomen
ontkoppelen afboeken
openzetten openstellen
verzenden versturen sturen
gerard inlogproblemen david
helpen assisteren
overzetten omzetten
vergroten uitbreiden
verstaan vermelden
inloggen aanmelden aanloggen
oplossen verhelpen
wijzigen aanpassen veranderen
invoeren invullen

12.5 Appendix E: Categorization of tickets

Categorylabel: ′work′, number of tickets: 15959
Categorylabel: ′laptop′, number of tickets: 15939
Categorylabel: ′wachtwoord′, number of tickets: 13890
Categorylabel: ′outlook′, number of tickets: 13341
Categorylabel: ′status navraag′, number of tickets: 12424
Categorylabel: ′dwr′, number of tickets: 11818
Categorylabel: ′printer′, number of tickets: 5698
Categorylabel: ′account′, number of tickets: 4745
Categorylabel: ′mail′, number of tickets: 4358
Categorylabel: ′blackberry′, number of tickets: 3936
Categorylabel: ′token′, number of tickets: 3641
Categorylabel: ′citrix′, number of tickets: 3146
Categorylabel: ′code′, number of tickets: 2722
Categorylabel: ′uem client′, number of tickets: 2351
Categorylabel: ′pc′, number of tickets: 2350
Categorylabel: ′beeldscherm′, number of tickets: 2227
Categorylabel: ′netwerk′, number of tickets: 2190

Categorylabel: ′taakbalk′, number of tickets: 310
Categorylabel: ′sap′, number of tickets: 304
Categorylabel: ′printing′, number of tickets: 296
Categorylabel: ′vpn′, number of tickets: 295
Categorylabel: ′p-direkt′, number of tickets: 286
Categorylabel: ′service′, number of tickets: 276
Categorylabel: ′usb′, number of tickets: 275
Categorylabel: ′topdesk′, number of tickets: 275
Categorylabel: ′afgehandeld′, number of tickets: 267
Categorylabel: ′bureaublad′, number of tickets: 252
Categorylabel: ′office′, number of tickets: 249
Categorylabel: ′tablet′, number of tickets: 246
Categorylabel: ′govroam′, number of tickets: 245
Categorylabel: ′kabel′, number of tickets: 241
Categorylabel: ′geluid′, number of tickets: 227
Categorylabel: ′mfc′, number of tickets: 225
Categorylabel: ′vip′, number of tickets: 215
Categorylabel: ′direct′, number of tickets: 214
Categorylabel: ′firefox′, number of tickets: 205
Categorylabel: ′ibabs′, number of tickets: 193

Categorylabel: ′telefoon′, number of tickets: 2157
Categorylabel: ′good′, number of tickets: 2016
Categorylabel: ′wifi′, number of tickets: 1825
Categorylabel: ′internet′, number of tickets: 1694
Categorylabel: ′scherm′, number of tickets: 1632
Categorylabel: ′postbus′, number of tickets: 1578
Categorylabel: ′ie′, number of tickets: 1576
Categorylabel: ′document′, number of tickets: 1540
Categorylabel: ′proxy′, number of tickets: 1447
Categorylabel: ′ww′, number of tickets: 1308
Categorylabel: ′schijf′, number of tickets: 1305
Categorylabel: ′toetsenbord′, number of tickets: 1296
Categorylabel: ′iphone′, number of tickets: 1231
Categorylabel: ′pst′, number of tickets: 1176
Categorylabel: ′sd′, number of tickets: 1139
Categorylabel: ′ontgrendelcode′, number of tickets: 1118
Categorylabel: ′muis′, number of tickets: 1090
Categorylabel: ′ipad′, number of tickets: 1064
Categorylabel: ′foutmelding′, number of tickets: 984
Categorylabel: ′agenda′, number of tickets: 912
Categorylabel: ′update′, number of tickets: 898
Categorylabel: ′gehoor′, number of tickets: 893
Categorylabel: ′flex2rijk′, number of tickets: 848
Categorylabel: ′toner′, number of tickets: 840
Categorylabel: ′computer′, number of tickets: 791
Categorylabel: ′monitor′, number of tickets: 785
Categorylabel: ′digidoc′, number of tickets: 747
Categorylabel: ′applicatie′, number of tickets: 742
Categorylabel: ′excel′, number of tickets: 724
Categorylabel: ′server′, number of tickets: 720
Categorylabel: ′port′, number of tickets: 670
Categorylabel: ′statusnavraag′, number of tickets: 661
Categorylabel: ′persoonlijke′, number of tickets: 658
Categorylabel: ′apps′, number of tickets: 641
Categorylabel: ′pdf′, number of tickets: 640
Categorylabel: ′hprm′, number of tickets: 604
Categorylabel: ′storing′, number of tickets: 572
Categorylabel: ′mobiel′, number of tickets: 561
Categorylabel: ′chromebook′, number of tickets: 524
Categorylabel: ′postvak′, number of tickets: 186
Categorylabel: ′samsung′, number of tickets: 182
Categorylabel: ′ssc-ict′, number of tickets: 180
Categorylabel: ′papier′, number of tickets: 172
Categorylabel: ′afdrukken′, number of tickets: 170
Categorylabel: ′vasco′, number of tickets: 165
Categorylabel: ′sim′, number of tickets: 162
Categorylabel: ′desktop′, number of tickets: 160
Categorylabel: ′exchange′, number of tickets: 158
Categorylabel: ′website′, number of tickets: 149
Categorylabel: ′ind′, number of tickets: 132
Categorylabel: ′powerpoint′, number of tickets: 126
Categorylabel: ′res′, number of tickets: 124
Categorylabel: ′abonnement′, number of tickets: 122
Categorylabel: ′szw′, number of tickets: 118
Categorylabel: ′hardware′, number of tickets: 117
Categorylabel: ′kiosk′, number of tickets: 113
Categorylabel: ′oracle′, number of tickets: 108
Categorylabel: ′smartphone′, number of tickets: 102
Categorylabel: ′u166′, number of tickets: 100
Categorylabel: ′domein′, number of tickets: 90
Categorylabel: ′spoed′, number of tickets: 76
Categorylabel: ′hp′, number of tickets: 67
Categorylabel: ′bes12′, number of tickets: 58
Categorylabel: ′ios′, number of tickets: 51
Categorylabel: ′explorer′, number of tickets: 50
Categorylabel: ′printen′, number of tickets: 49
Categorylabel: ′contacten′, number of tickets: 48
Categorylabel: ′pro′, number of tickets: 41
Categorylabel: ′stick′, number of tickets: 25
Categorylabel: ′leenlaptop′, number of tickets: 21
Total number of tickets categorized: 179113

Categorylabel: ′werkplek′, number of tickets: 513 Categorylabel: ′rijksportaal′, number of tickets: 495 Categorylabel: ′defect′, number of tickets: 493 Categorylabel: ′simkaart′, number of tickets: 479 Categorylabel: ′bes′, number of tickets: 466 Categorylabel: ′data′, number of tickets: 465 Categorylabel: ′spam′, number of tickets: 455 Categorylabel: ′software′, number of tickets: 446 Categorylabel: ′adobe′, number of tickets: 436 Categorylabel: ′beeld′, number of tickets: 426 Categorylabel: ′umts′, number of tickets: 406 Categorylabel: ′toestel′, number of tickets: 402 Categorylabel: ′password′, number of tickets: 398 Categorylabel: ′dwr-next′, number of tickets: 377 Categorylabel: ′mappen′, number of tickets: 359 Categorylabel: ′huis′, number of tickets: 341 Categorylabel: ′follow′, number of tickets: 339 Categorylabel: ′windows′, number of tickets: 326 Categorylabel: ′vodafone′, number of tickets: 317 Categorylabel: ′profiel′, number of tickets: 314	