Towards Automated Classification of Aircraft Maintenance Documentation

EASA AD No : 2013-0277R1

EASA	AIRWORTHINESS DIRECTIVE
	AD No.: 2013-0277R1 **Date: 04 December 2013** Note: This Airworthiness Directive (AD) is issued by EASA, acting in accordance with Regulation (EC) No 216/2008 on behalf of the European Community, its Member States and of the European third countries that participate in the activities of EASA under Article 66 of that Regulation.

This AD is issued in accordance with EU 748/2012, Part 21.A.3B. In accordance with EC 2042/2003 Annex I, Part M.A.301, the continuing airworthiness of an aircraft shall be ensured by accomplishing any applicable ADs. Consequently, no person may operate an aircraft to which an AD applies, except in accordance with the requirements of that AD, unless otherwise specified by the Agency [EC 2042/2003 Annex I, Part M.A.303] or agreed with the Authority of the State of Registry [EC 216/2008, Article 14(4) exemption].

Type Approval Holder's Name :	Type/Model designation(s) :
AIRBUS	A318, A319, A320 and A321 aeroplanes

TCDS Number : EASA.A.064

Search

Registration mark
CS-TJE

Topic
Wings (ATA 57)

Aircraft Manufacturer
Airbus

Aircraft Type
A321-211

Relevant Documents:

2013-0277R1
Effective: 10/12/2013

2012-0223
Effective: 06/11/2012

2012-0032R1
Effective: 08/12/2015

2011-0121R1
Effective: 27/07/2011

TU Delft

01-12-2016

L. Scherp

This page is intentionally left blank.

Towards Automated Classification of Aircraft Maintenance Documentation

By

L. Scherp

in partial fulfilment of the requirements for the degree of

Master of Science
in Aerospace Engineering

at the Delft University of Technology,
to be defended publicly on Friday December 16, 2016 at 2:00 PM.

Supervisor:	Dr. ir. W.J.C. Verhagen	
Thesis committee:	Prof. dr. R. Curran,	TU Delft
	Dr. ir. W.J.C. Verhagen,	TU Delft
	Dr. ir. A. Sahai,	TU Delft
	Ir. H. Koornneef,	TU Delft

TU Delft

This page is intentionally left blank.

Preface

This thesis is the last step towards finishing the Master of Science (MSc) degree in Aerospace Engineering at the TU Delft. The specialization chosen in this degree is Air Transport Operations, which is a focus on the operational side of aircraft, airlines and airports. After completing all courses, exams and an internship, this thesis is the final research project of the curriculum. It is carried out in the aircraft maintenance domain, which deals with the repairs, inspection, modification and overhaul of aircraft. To be more specific, the research is performed to find a recommended approach for creating a system that can process digital maintenance documentation that can be provide relevant documents to an aircraft maintenance technician during work. As it is an exploratory research in this area, the thesis is titled 'Towards Automated Classification of Aircraft Maintenance Documentation'.

The thesis is an individual assignment, but doing the research and writing this report is performed with the help of the people around me. Via this way, I would like to thank my supervisors, family and friends for their support along the way.

First and foremost, starting this thesis project would have not been possible without the aid of my parents for many years. In this way, I would like to express my gratitude once again for the support and motivation while pursue my degrees.

Second, I would like to thank my daily supervisor, Wim Verhagen, for this academic guidance and support during the thesis project. The discussions and feedback during our meetings have helped greatly to create the report and test system. Additionally, I would like to thank Ricky Curran, holder of the Air Transport Operations chair, for the valuable feedback during the evaluation sessions. It especially helped to keep a critical view on the scope of the project.

Third, a massive thanks to my unofficial supervisor, Hemmo. Continuing his work has been a great experience and the many sparring sessions, discussions, feedback and fun along the way has made this thesis my best university experience. Additionally, I would like to thank Arnold and Tomas for the inspiration, fun and helpful feedback, which made me accomplish more than I expected at the start of this project.

Finally, I would like to thank my girlfriend Lieke for her support during the thesis. Thank you for listening when I had an error at random moments and the motivation and fun along the way. Without you, this experience would not have been the same.

L. Scherp
The Hague, December 2016

This page is intentionally left blank.

Abstract

In the field of aircraft operations, maintenance is crucial for ensuring continued aircraft safety and airworthiness. The maintenance is performed either scheduled as a preventive task, or unscheduled, when an unexpected failure occurs on the aircraft. If immediate maintenance action is required, the unscheduled tasks are executed during the turnaround time of the aircraft to ensure airworthiness while minimizing the impact on the aircraft flight time. The limited time available during turnaround causes the aircraft maintenance technician (AMT) to feel a pressure to perform as fast as possible, to avoid costly flight disruptions. As AMTs still use paper-based documentation and have limited access while on a task, finding the right information in this limited time can take up to 15-30% of the task. In practice this means that during the execution of the task, the AMT must make a difficult tradeoff between either searching the right documents which means spending valuable maintenance time, or not using documentation and increasing the risk of errors. To avoid this, the search time should be reduced. This can be achieved by a switch to digital documentation and using a system to automate the information retrieval process, so that the AMT can focus on the actual repair of the problem instead of searching for information. However, the functionality of such a system is not available in the aircraft maintenance literature. Therefore, this thesis focusses on finding suitable functionalities and techniques to use in such an automated documentation system. The main goal is to formulate recommendations for how to create a system that can automatically identify and extract relevant information from maintenance documentation to aid the AMT during unscheduled tasks.

As the literature in the aircraft maintenance domain is very limited towards these automated systems, first an overview of possible approaches was created based on techniques and functionalities used in other domains. Six major research domains were identified that focus on the retrieval and extraction of information from text. These domains are: Natural Language Processing (NLP), Information Retrieval (IR), Information Extraction (IE), Text Mining (TM), Machine Learning (ML) and Knowledge Discovery in Text (KDT). Using the findings obtained from these domains, a novel theoretical functional flow diagram (FFD) was created in this thesis based on an aggregation of the information in the six identified domains. The developed FFD describes the functionalities that are necessary in an automated documentation system and covers the pre-processing of documents and information delivery to the user. Additionally, from the six domains a set of techniques was defined that can be used in the functions in the FFD. Three groups of techniques are identified in the FFD: document processing, pre-processing of document contents and the ad-hoc processing of a search request. For document processing, techniques are found such as PDF2Text and tokenization, which enable the processing of raw documents to plain text. The pre-processing techniques are used for classification and clustering of the documents, the techniques in this group range from simple Boolean techniques to more advanced Machine Learning techniques such as the Naive Bayes, k-Nearest Neighbors (kNN) or fuzzy clustering techniques. Lastly, the ad-hoc processing can be performed by the Boolean technique and advanced techniques such as Latent Semantic Indexing.

Based on this theoretical foundation a generic automated documentation system was developed. It can work with any textual data source, technique or information request, so that it can be used in many contexts. This novel system consists of three main elements: document processing, testing of techniques (pre-processing) and a live search. The document processing functions are used to convert raw pdf to tokenized documents, which can then be used by the testing functions. The testing of techniques is based on a test setup that can be provided on a web client user interface. A test setup is a combination of a technique, document sample and the information that should be identified

and extracted from the documents. The test results can be stored and accessed by the live search system, which demonstrates the capabilities of such a system: provide an information query and retrieve relevant documents to that query. This live search is also available on a web client interface.

This generic documentation system was extended to work with specific types of data. In this thesis, it was chosen to select Airworthiness Directives (ADs) as the data source as these are available in the public domain and in sufficient quantity to be able to perform ML techniques. A total of 14008 AD documents were obtained from an EASA AD tool, from which 10571 were readable by a pdf text extraction script and had meaningful content. To compare the performance of techniques on this dataset of 10571 ADs, a use case was defined that determines which information is relevant to extract from the documents: given an aircraft registration mark and a topic, identify the ADs that are applicable / relevant. To implement this use case, the TAP aircraft fleet was used to have tangible data to search for in the documents. Based on a selection of aircraft from the TAP fleet, five relevant document properties are determined that must be extracted from the ADs to fulfil the use case: Document Publisher, Document Language, Document Type, Document Superseded and Document Aircraft Manufacturer. For each of these properties, three techniques were tested: the Boolean technique and two Machine Learning techniques: Naive Bayes and kNN.

The performance assessment of the techniques was done based on the F-score, which is a standardized combined score of precision and recall. The Naive Bayes technique performed best on the Document Publisher and Language properties, with scores of 98.6% and 100%. Additionally, the Boolean technique scored the highest on the Document Type and Document Aircraft Manufacturer properties with a F-score of 97.1% and 93.1% respectively. Lastly, the Document Superseded property was best extracted by the kNN technique with a score of 67.0%. Overall, almost all properties could be extracted with a very high F-score (>93%). This is a positively surprising result, as lower scores for more complex tasks were expected based on literature, which estimates scores of about 80 to 85% for complex tasks.

It was found that each of the three considered techniques have their specific benefit depending on the document property. The Boolean method performed best on properties that can be captured with a single word in the text. However, if there is no single keyword describing the property, such as Language or Publisher, the Machine Learning methods perform significantly better. Between the Machine Learning methods considered, it can be concluded that the Naive Bayes method is better if more words in a document are relevant for the classification of a document. On the other hand, the kNN technique is better in distinguishing relevant and irrelevant words. It outperformed Naive Bayes if there was only a set of relevant words in the documents.

It is concluded that for identifying and extracting information from ADs, the techniques to apply must be selected based on the property to get the best performance. If another type of maintenance documentation would be used in this system, such as the Aircraft Maintenance Manual, the process of finding the best technique for each relevant property should be repeated. It means that for each type of document and property in the documents, the system must be specifically designed and tested before a selection can be made. By obtaining the results for the case of ADs and creating the prototype system that can be modified for other types of maintenance documentation, a recommended approach is established for how to design an automated documentation system to aid the AMT during unscheduled tasks.

Table of Contents

List of Figures

List of Tables

List of Acronyms and Abbreviations

AD	Airworthiness Directive
AMT	Aircraft Maintenance Technician
ATA	Air Transport Association
DGAC	Directorate General for Civil Aviation
EAD	Emergency Airworthiness Directive
EASA	European Aviation Safety Agency
FAA	Federal Aviation Administration
FFD	Functional Flow Diagram
FP	False Positives
FN	False Negatives
FTP	File Transfer Protocol
HMM	Hidden Markov Method
IATA	International Air Transport Association
IDF	Inverse Document Frequency
IE	Information Extraction
IR	Information Retrieval
JS	JavaScript
KDD	Knowledge Discovery in Databases
KDT	Knowledge Discovery in Text
kNN	k-Nearest Neighbors
ML	Machine Learning
NER	Named Entity Recognition
NLP	Natural Language Processing
OWL	Web Ontology Language
SB	Service Bulletin
SVM	Support Vector Machines
TAP	Transportes Aéreos Portugueses
TC	Transports Canada
TF	Term Frequency
TP	True Positives
TN	True Negatives
TM	Text Mining
VNC	Virtual Network Computing
VPS	Virtual Private Server
XML	eXtensible Markup Language

1. Introduction

In the field of aircraft operations, maintenance is crucial for ensuring continued aircraft safety and airworthiness (McDonald et al. 2000). Maintenance is performed either as a scheduled or unscheduled task (Kinnison & Siddiqui 2012). Scheduled maintenance is a preventive task and executed regularly when aircraft components are checked, repaired or replaced. Unscheduled maintenance is performed in case of an unexpected event, such as component failure during flight. If possible, these unexpected errors have to be repaired during the turnaround time of the aircraft to ensure airworthiness while minimizing flight disruptions (Kinnison & Siddiqui 2012).

The limited time available during turnaround and the lack of information on the specific problem causes the aircraft maintenance technician (AMT) to often feel a tension between safety and economic interests (Atak & Kingma 2011). If AMTs are unable to perform their task during the turnaround time, flight delays or disruptions can occur (Civil Aviation Authority 2002). The AMT has the difficult task of obtaining the right information and fixing the problem in this limited time, for which the AMT can either rely on his own knowledge and experience or consult maintenance documentation. As AMTs still mainly use paper-based documentation during the execution of maintenance tasks (Lee et al. 2008), finding the right information in those documents can take between 15 and 30% of a task (Lampe et al. 2004; Taylor 2008). In practice, this means that during the execution of the task the AMT has to make a difficult tradeoff between either finding the right documents and spending valuable maintenance time on this search, or not using documentation and increasing the risk of errors or mistakes (Zafiharimalala et al. 2014).

To avoid this tradeoff between undesirable options (delays vs. errors), the search time should be reduced so that the AMT has more time to complete his task while using the right documents to ensure a correct process. This can be achieved by moving to digital documentation and using a system to automate the information retrieval process, so that the AMT can focus on the actual repair of the problem instead of searching for information (Zafiharimalala et al. 2014). Such an automated system would be able to identify and extract relevant sections from the documentation and provide these to the AMT during the execution of an unscheduled task. This move toward digital documentation is also supported from a regulatory perspective and fits in the IATA vision: “To simplify maintenance operations by incorporating paperless technologies, thereby facilitating regulatory compliance and enabling new processes to reduce costs” (IATA 2013). Although a vision is clearly formulated, the functionality of such a system is not available in aircraft maintenance literature (Lee et al. 2008; Verhagen & Curran 2013). As a step towards this goal, this thesis answers the following research question:

What is the recommended approach to identify and extract relevant information from digital maintenance documentation to aid a maintenance technician during unscheduled tasks?

An approach for an automated documentation system is defined as a combination of functionalities and techniques to identify and extract the relevant information. As there are no automated documentation systems available in literature, a new system is created from scratch. It is first designed as a generic system that can perform any document processing task. In this way, this novel prototype system can process documents and test different techniques for classification of these documents. Additionally, the testing can be performed on different document properties, techniques and other specific settings. After establishing the generic system, it is extended to work with the data

and techniques in this thesis. To compare the different techniques for a similar set of documents, it is chosen to use Airworthiness Directivess (ADs) as test documents.

The functionalities and techniques required in this prototype system are obtained from six major research domains that focus on the retrieval and extraction of information from data. These domains are: Natural Language Processing (NLP), Information Retrieval (IR), Information Extraction (IE), Text Mining (TM), Machine Learning (ML) and Knowledge Discovery in Text (KDT). A novel theoretical functional flow diagram (FFD) is created in this thesis based on an aggregation of the information in these six domains. This FFD describes the functionalities that are necessary in an automated system and covers the pre-processing of documents and information delivery to the user. Additionally, from these domains a set of techniques is defined that can be used for the functions in the FFD. These techniques range from simple Boolean techniques to more advanced Machine Learning techniques such as the Naive Bayes technique.

Based on the theoretical techniques and using the system, a comparison of these techniques is made to formulate a recommended approach to identify and extraction relevant information from ADs. The research objective encompasses all elements necessary to make this comparison. The objective is to identify theoretical approaches, create a prototype system that serves as a test environment, evaluate possible approaches in this system and lastly formulate recommendations for the design of an automated documentation system.

This thesis is structured accordingly. First, Chapter 2 presents a literature review of automated documentation systems, providing an overview of the state-of-the-art in the aircraft maintenance domain and six other domains. Chapter 3 then describes the methodology used for testing the different techniques, which shows details on the development of the system and the different elements of the test setup such as the data and assessment criteria. Subsequently, Chapter 4 demonstrates the results of the tests for the established test setups. Finally, Chapter 5 discusses the conclusions, research contributions, limitations and recommendations of this thesis.

2. Literature Review of Automated Documentation Systems

This chapter describes the literature regarding the use of automated systems for identification and extraction of relevant information from documents. More specifically, top-level systems and techniques used in these systems are discussed based on literature from the aircraft maintenance domain and other domains. The chapter outline is visualized in Figure 2.1. After this introduction, an overview of the state-of-the-art in the aircraft maintenance domain is provided in section 2.1. This is followed by indicating the research gaps found in the aircraft maintenance domain in section 2.2, which mainly boils down to the lack of systems and techniques related to maintenance documentation in this domain. Therefore, an overview of the state-of-the-art in generic domains is presented in section 2.3. These generic domains are Natural Language Processing (NLP), Information Retrieval (IR), Information Extraction (IE), Text Mining (TM), Machine Learning (ML) and KDT (Knowledge Discovery from Text). A novel functional flow diagram is then created based on all functionalities described in these six domains. The functionality is divided in three groups of processing: document pre-processing, ad-hoc processing and planned processing.

Having established the functionality required for an automated system for the identification and extraction of information from documents, the next section focus on the available techniques to perform these functions. First, section 2.4 gives an overview of assessment criteria that are used to measure the performance of techniques. Subsequently, sections 2.5, 2.6, 2.7 and 2.8 describe the document pre-processing techniques, ad-hoc processing techniques, planned processing techniques based on text classification and planned processing techniques based on text clustering respectively. These techniques are afterwards synthesized in section 2.9. After having created this synthesis, the application of these techniques in other specific domains is briefly discussed in section 2.10. In contrast with the aircraft maintenance domain, there are many applications of these techniques in the medical, biology and manufacturing domain. The last section of this chapter, section 2.11, is a conclusion to this chapter. It is a discussion of the findings in the generic and specific domains and it elaborates on how the identified research gaps are covered with this work.

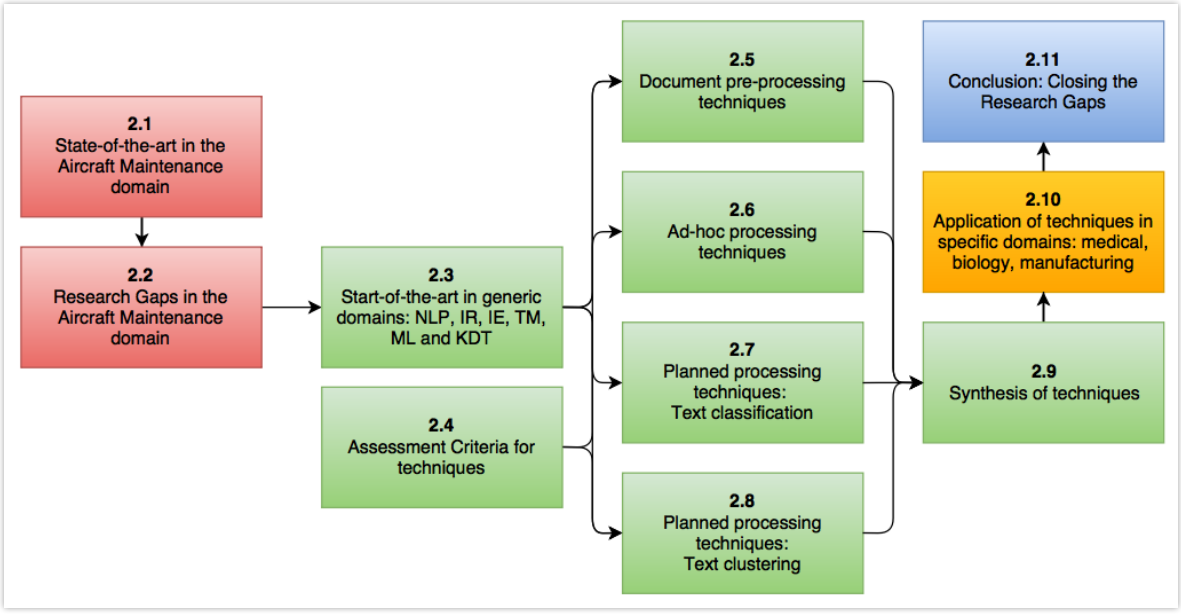


Figure 2.1: Outline of the Literature Review chapter

2.1. State-of-the-art in the Aircraft Maintenance Domain

Aircraft maintenance, while critical for ensuring the safety of aircraft, is an under researched area when compared to other aircraft related research domains (Atak & Kingma 2011). The available literature in the aircraft maintenance domain related to maintenance information systems is very limited (Lee et al. 2008; Verhagen & Curran 2013). In the generic maintenance management domain also very little research is performed related to maintenance information systems, as most of the literature is related to maintenance optimization, maintenance techniques and performance measurement (Garg & Deshmukh 2006). More specifically, it is found that most line maintenance processes still rely on paper-based documents rather than using digital systems (Lampe et al. 2004; Verhagen & Curran 2013).

The relevant available literature related to systems, methods and techniques can be grouped under the term eMaintenance. It can be defined as excellent, electronic or a combination of efficient + effective + enterprise Maintenance (Muller et al. 2008). The term has emerged early 2000 in the literature and is defined as the "monitoring, collection, recording and distribution of real-time system health data, maintenance-generated data as well as other decision and performance-support information to different stakeholders independent of organization or geographical location, 24h a day, 7 days a week" (Candell et al. 2009). In a broader fashion, eMaintenance is concerned with the use and integration of IT solutions in the maintenance domain (Levrat et al. 2008). Examples are e-technologies, e-monitoring, e-diagnosis, e-prognosis, etc.

Despite the limited research available in this domain, there are incentives to develop eMaintenance platforms that cover (parts of) the research objective of this thesis. The next subsections provide an overview of relevant platforms found in literature.

IDS project (1997)

The earliest mention in the literature of an applied system that uses digital maintenance documentation is the IDS (Integrated Diagnostic System) project and is developed in the context of the maintenance operations of Air Canada. It is aimed at creating a single point of information for the line maintenance technician where he could access all kinds of maintenance related information such as maintenance documentation (Wylie et al. 1997). The core process behind the IDS is filtering and aggregating all information related to a maintenance task from various sources such as manuals, heuristics and historical data. The techniques used to perform these actions are rule-based and case-based reasoning, e.g. providing the Trouble Shooting Manual (TSM) of an aircraft. This project was still a prototype and future literature did not describe additional developments on this platform. In short, it is a simple approach to identify and extract information by creating rules for specific information, but it is a time-consuming task if the amount of information increases.

FTRAN/SNAGIE method (1999; 2001)

As a follow-up to the IDS project, Farley (1999; 2001) describes the SNAGIE method, which is used to extract and digitalize written maintenance log book information. SNAGIE stands for Snag Information Extractor and it is described as a group of NLP techniques such as grammatical analysis and semantic interpretation evaluation of free-text repair action notes (FTRAN). Each entry in the maintenance logbook is called a snag and these techniques are applied to extract relevant information in each entry and the actions performed, such as pieces of equipment and related actions to that equipment (Farley 1999). Its functional diagram is displayed in Figure 2.2, with the core elements the lexicon, analysis tools and interpretations evaluation.

The main work is done on creating a lexicon to match documents with, by building on the Alvey Natural Language Toolkit (ANLT) which is a general dictionary that is not specific to the maintenance domain. As with the IDS project, this project is in a prototype phase when it is described in the article and no further developments could be found. No results of extracting information from the log book entries are discussed in the papers. Some interesting text processing techniques are used (e.g. parsing and named entity recognition) and these are discussed in section 2.5.

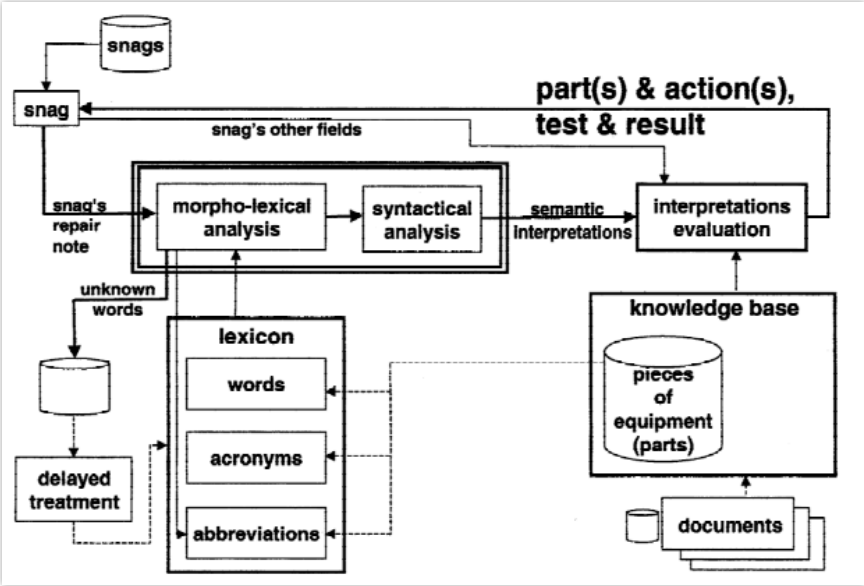


Figure 2.2: Functional diagram of the SNAGIE method (Farley 1999)

ICAS platform (2003)

As the term eMaintenance emerged early 2000s, the ICAS platform is regarded as one of the first eMaintenance platforms developed (Levrat & Iung 2007). The Integrated Condition Assessment System (ICAS) is a commercial condition-based maintenance (CBM) system that aims to improve the reliability and availability of equipment (Hogan et al. 2003). The focus of the platform is on the interactions between the several subsystems such as data acquisition, expert analysis and historical information, rather than specific information related to using documentation in such a system. Therefore, it can be described as a generic IT system to improve the maintenance process, but no relevant tools or techniques are described in the context of the research objective.

ITEA European project: PROTEUS system (2002-2005)

Besides industry-related projects such as the Air Canada related IDS project, also some European projects were created that have elements of using digital maintenance documentation. The ITEA European project ran from 2002 to 2005 and it was aimed at developing a software platform to integrate software modules related to diagnostic and remote maintenance and as a solution the PROTEUS system was developed (Levrat & Iung 2007). To be more specific, the PROTEUS system is a data integration platform that is able to automatically exchange information between the equipment manufacturer, integrator and end-user (Bangemann et al. 2004). To get the right information to the right person, the system uses many artificial intelligence tools (e.g. hidden Markov Models or fuzzy systems) to be able to perform diagnosis or prognosis (Dechamp & PROTEUS WP2 Team 2004). The PROTEUS system covers some techniques that could be used for fulfilling the research objective, these are discussed in section 2.7 and 2.8. The overall PROTEUS system proves to be an interesting governing IT system for the maintenance process, but besides the mentioned techniques it is very broad and generic.

CASIP platform (2004)

The Computer Aided Safety and Industrial Productivity (CASIP) platform is an eMaintenance system that aims to integrate subsystems related to remote monitoring, diagnosis and tele-maintenance (Léger 2004). The CASIP platform covers many subsystems that enable the remote access from a centralized database (Advitium) to the Enterprise Resources Planning (AdoniX) system or the Computerized Maintenance Management System (EmpaciX), etc. The literature mainly describes the interactions between the different subsystems of the platform and therefore does not provide relevant techniques or methods to be applied at the task at hand. Additionally, it is more focused on remote maintenance and diagnosis than accessing relevant maintenance documentation as it is assumed that the relevant information is already stored and accessible in the central database.

TEMIC platform (2004)

A similar platform as the CASIP platform is the TEMIC (Tele-Maintenance) platform that enables users to connect information systems with an application on their PDA (Brahma et al. 2006). It also allows to collect data from remote sensors, report on activities and perform diagnostic and other collaborative actions via the network with other experts (Garcia et al. 2004). Relevant techniques are like the PROTEUS system: using hidden Markov models neural networks to detect anomalies in the sensor data to know when equipment or tools should be maintained. These techniques will be discussed later in this literature review (see section 2.7 and 2.8). The TEMIC platform by itself is a governing IT system and therefore not relevant for the research objective.

Tinker Air Force Base system (2004)

At the Tinker Air Force Base, a portable device is developed commercially by UGS and Intel that is able to wirelessly access maintenance records and technical manuals (Thilmany 2004). Additionally, 3D CAD models of the items that need to be repaired are displayed. Unfortunately, no additional detailed information about the system is mentioned.

TELMA platform (2007)

So far, eMaintenance systems have been discussed that could be used as a governing IT system to a maintenance task. The TELMA platform provides a training platform in the field of maintenance, tele-maintenance and e-maintenance (Levrat & Iung 2007). It is an academically pushed platform that could be used for supporting both academics as technicians to learn to work with eMaintenance systems. The maintenance process is simulated, but no specific techniques or tools are mentioned related to the access and retrieval of relevant information from maintenance documents.

TATEM European project (2003-2008)

The TATEM project (Technologies and Techniques for New Maintenance concepts) demonstrated the need for technologies to increase aircraft operability by reducing the occurrence of unscheduled maintenance and the time and cost of scheduled maintenance (Taylor 2008). The scope of this project covered health monitoring, integrated data management, maintenance planning, and mobile maintenance. It is found that a process-oriented maintenance approach is required and subsequently supporting software prototypes are developed. The prototypes related to retrieving task-specific information from documents did not provide contextualized documentation and no validation was performed of the tools. Unfortunately, no relevant techniques are found, so it cannot be applied in this thesis.

DYNAMITE European project (2005-2009)

As some of the previous platforms, The DYNAMITE (Dynamic Decisions in Maintenance) European project also focusses on creating a wireless infrastructure for maintenance processes (Holmberg 2005). However, additionally it aims to create new devices to improve the decision systems related to maintenance processes. The work in this project was grouped in 3 parts: condition monitoring sensors, mobile & wireless devices and technologies and economic studies in the context of maintenance strategy (Jantunen & Gilabert 2010). It focused mainly on a top-level IT system and although interesting mobile applications and an infrastructure are discussed, no relevant techniques or tools are found that can be used in the context of this work.

HILAS European project (2005-2010)

The European project HILAS (Human Integration into the Life-cycle of Aviation Systems) focused more on the human element in the maintenance process, more specifically it aims to capture the knowledge generated by humans when operating a maintenance system so that it can be used in the design of more effective systems or technologies (McDonald 2010). A prototype is developed with the functionality to select a subset of task-specific documentation, but no sections from documents can be extracted. This can be inefficient, for example in case of an Aircraft Maintenance Manual, which consists of thousands of pages. Moreover, no techniques are described in detail and there is no validation of the prototype. Additionally, the prototype was not able to work with legacy documentation.

Ontology-based methods (from 2007)

Ontologies are often used as an element of the prototypes and methods described in the previous sections, but they are rarely discussed in detail (Malin & Throop 2007; Verhagen & Curran 2013). The first paper that describes how to develop an ontology in the aerospace domain is by Malin & Throop (2007), where it is used to extract useful information from text to determine failure modes and for effects analysis. It is described as a snapshot instead of a fully developed ontology. They propose to use the ontology in combination with their Reconciler tool that can be used for semantic text analysis (more information on semantic text analysis in section 2.3). The authors mention that the challenge in developing an ontology is that domain-specific knowledge evolves and therefore the ontology should evolve as well. Amardeilh et al. (2013) also describe the difficulty in maintaining an ontology and propose the use of a system called Virtuoso to manage and maintain an ontology. Their system is a prototype to automatically add new information to ontologies, but it is only tested limited and should be further developed.

Verhagen & Curran (2013) demonstrate the use of an ontology in aircraft maintenance task support as a proof of concept, more specifically storing relevant information from maintenance related documentation in an ontology. Besides using the ontology to semantically annotate information and for storing the information, there are no specific techniques mentioned that can be used for fulfilling the research objective. In a similar fashion, Gargiulo et al. (2014) propose the use of an ontology and taxonomy in the aerospace domain to allow semantically searching information in aerospace documents. It is however not specifically tailored to maintenance documentation. The tool is part of the SIA portal (Sistema Informativo Aerospaziale) and uses a Bayesian model (statistical method for comparing similarity between datasets) for classification of text by using the information in the ontology and taxonomy. The application of a Bayesian model is described later in this chapter, see section 2.7.

2.2. Research Gaps in the Aircraft Maintenance Domain

The literature in the aircraft maintenance domain mainly focusses at performance measurement and the optimization of maintenance processes. It is found that some top-level IT systems are available in the aircraft maintenance context, but few describe the identification and extraction of information from maintenance documents. A few techniques were found that are used in the maintenance domain, such as rule-based techniques, case-based techniques, neural networks, hidden Markov models and Bayesian models. Despite being relevant techniques, these were often not used in the context of identifying and extracting relevant information from documents. Therefore, two main research gaps are identified in the aircraft maintenance domain: almost no literature is available related to automated systems for maintenance documentation and likewise for approaches and techniques that can be used in these systems.

While academic literature is limited, from an industrial perspective it is found that airlines and manufacturers dedicate resources to further developing digital documentation systems. Detailed information is often not available, but to give a few examples of companies working on this task, Lufthansa Technik is developing systems to move from paper-based maintenance documents to electronic documents (Lufthansa Technik AG 2016). Additionally, AIRBUS is working with the tool AirN@v, which is a suite of to create, update, and revise all the maintenance and digital documentation for aircraft (Airbus 2016), but this tool is limited in its functionality. In a similar fashion, Emirates Engineering has created a Technical Resource Centre, where they store their electronic maintenance documentation. Both airlines and aircraft manufacturers increase the use of digital information, but it is just a first step towards using digital documents during the maintenance task.

Additionally, the discussed platforms and systems have specific limitations. Tretten & Karim (2014) put a critical note at eMaintenance by illustrating that existing eMaintenance solutions often suffer from an insufficient level of usability and quality of information, which in itself leads to errors and mistakes by the users of the solutions. Jardine et al. (2006) mention the lack of high-quality data, lack of good communication between theory developers and practitioners, lack of validation approaches and difficulty of implementation due to frequent change of design and technologies. Muller et al. (2008) and Aljumaili et al. (2012) mention the quality of the documentation to be an issue, for example difficult to use legacy documentation. Farley (2001) mentioned that validation is a key missing element in the development of these systems. It is recommended to validate a system by domain experts when a system is developed and results can be generated, however finding experts has so far proven to be difficult if the system is not designed in the context of a maintenance company (Farley 2001). Despite the difficulty in creating an automated contextualized documentation system, many authors note the importance of such a system, e.g. Lee et al. (2008); Verhagen & Curran (2013); Tretten & Karim (2014).

In the context of this thesis, which is centered around finding a recommended approach to contextualize maintenance documentation, the research gaps identified should be addressed to have an overview of how such an automated system should work and which techniques or approaches are best applied in such a system. To tackle these research gaps, the next section focusses on describing generic domains related to the identification and extraction of information from documents, stepping away from the aircraft maintenance domain. By identifying relevant literature in these generic domains, a first step is made to bridge the research gaps in the aircraft maintenance domain.

2.3. State-of-the-art in generic domains: NLP, IR, IE, TM, ML & KDT

The last section demonstrated two research gaps in the aircraft maintenance domain, one of which is the lack of literature related to techniques that can be used for the extraction and identification of information in aircraft maintenance documents. This section describes the state-of-the-art in the generic research domains related to this task by systematically describing relevant research domains. Six major generic research domains can be identified that cover functions for the identification and extraction of information from documents: Natural Language Processing (NLP), Information Retrieval (IR), Information Extraction (IE), Text Mining (TM), Knowledge Discovery in Text (KDT) and Machine Learning (ML). The next subsections (2.3.1 – 2.3.6) describes these domains top-level, i.e. the state-of-the-art of literature and major tasks that can be performed with functionality available in each domain, without going into detail in the techniques available in each domain. Finally, in subsection 2.3.7 these functions are synthesized in a novel functional flow diagram covering all domains.

2.3.1. Natural Language Processing (NLP)

A system that automatically identifies and extracts useful information in a text must be able to 'read' documents to function. The governing research domain for processing text is called Natural Language Processing (NLP). It is defined as a range of computational techniques to automate the representation and analysis of languages (Cambria & White 2014). The first work in NLP dates back to the '50s and originates from a mix between the artificial intelligence and linguistics domains (Nadkarni et al. 2011). Artificial intelligence is defined as the study of systems that think/act like humans (Russell et al. 1995) and linguistics is the scientific study of human language (Himmelmann 1998).

NLP covers a broad spectrum of syntactic and semantic techniques for the analysis of human language texts with the ultimate goal to transcend from processing to the understanding of human language (Collobert et al. 2011; Cambria & White 2014). The research related to the techniques can be mapped over time as is visualized in Figure 2.3. The first curve that can be identified is the Syntactics curve that contains word-based tasks such as keyword spotting and more advanced methods based on statistics. Additionally, commonly used NLP syntactic techniques are breaking down a text to sentences (sentence tokenization) or creating a parse tree from a text (Dubitzky & Azuaje 2004). These NLP techniques are often used to pre-process documents so that information in the text can be easier identified and extracted.

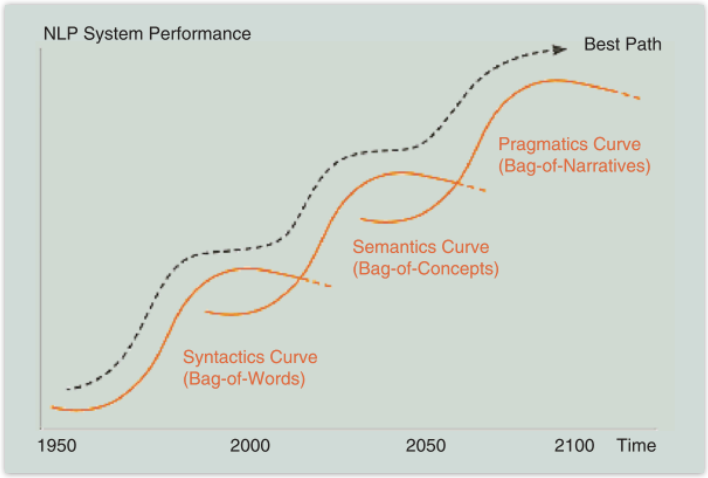


Figure 2.3: Evolution of NLP research through three eras (Cambria & White 2014)

The second curve that starts around 2000 is the Semantics Curve that focusses on exploiting the semantics and sentics of multiple words, where semantics is related to the contextual meaning of words and sentics describes the affective information associated with words that humans use for common-sense reasoning (Cambria & Hussain 2012). Approaches that use semantics are often based on machine learning techniques and large taxonomies and ontologies of words and their relations (Fernández et al. 2011). Despite the trend of growing semantics research, most of the systems used for retrieving documents and extracting information are based on syntactic methods (Nadkarni et al. 2011). A Pragmatics Curve is expected in the future that focusses on the meaning of natural language, but first the semantic techniques should be further developed.

2.3.2. Information Retrieval (IR)

The NLP techniques are used for tasks in many research domains related to identification and extraction of relevant information from texts. One of the first domains focused on access of information in texts is called the Information Retrieval (IR) domain. It is defined as “the process of retrieving a ranked set of relevant documents from a corpus based on a user’s stated information need” (Small & Medsker 2014). The goal of methods in this domain is to help a user find relevant documents in a set of documents, so creating a relevant subset of documents to a user’s query. Early work is performed in the ‘60s in the Cranfield studies (Cleverdon 1960) that experimented with the use of NLP techniques in the indexing of documentation and a further boost to the research domain is given in the ‘90s by a series of TRECs (Text Retrieval Conferences) (Voorhees 2007). Originally the IR domain was distinct from NLP but they converged somewhat as many NLP techniques are used in IR (Nadkarni et al. 2011).

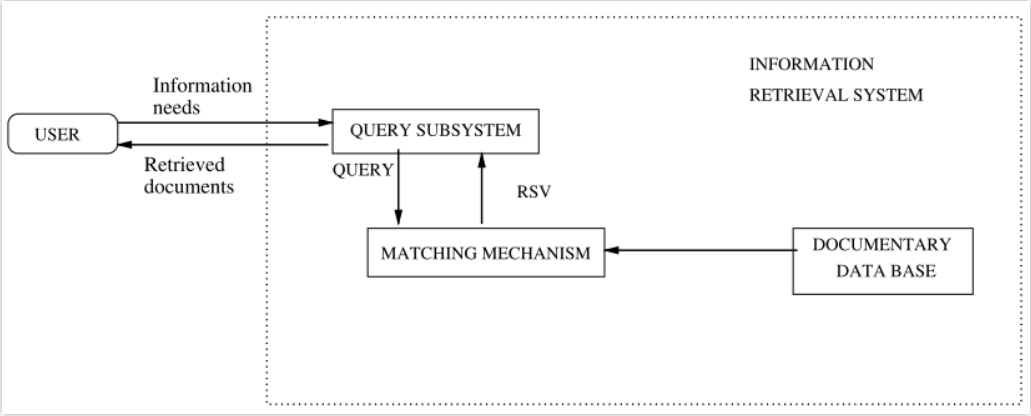


Figure 2.4: A basic IR system (Cordón et al. 2003)

A typical IR process consists of four phases: indexing, query formulation, comparison and feedback (Lewis 1991). The indexing phase is used to convert a set of ‘raw’ documents or texts to a suitable representation for further processing such as displaying the documents as vectors of words. The query formulation phase is done by the user of the system, determining the query for the information request. The next phase, comparison, matches the indexed documents with the query and orders the documents based on their relevance to the query. The last phase of a typical IR process is the feedback phase, where the user evaluates the retrieved information by the system and reformulates the query if necessary. Another way of viewing the IR processes is visualized in Figure 2.4, where it can be seen that a user query (information need) is sent to the query subsystem that interacts with the matching mechanism to retrieve the relevant documents for the user (Cordón et al. 2003). The process steps are similar to the ones depicted by Lewis (1991), but named differently or combined.

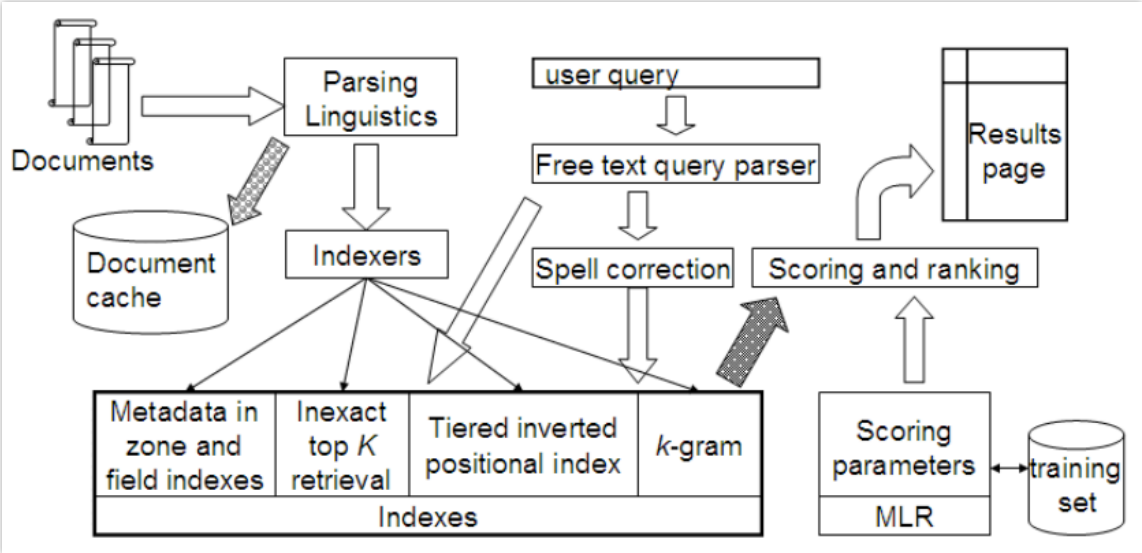


Figure 2.5: A complete IR search system (Manning et al., 2009)

As yet another way of viewing IR, Manning et al. (2009) describe a full search system that incorporates NLP and IR as is visualized in Figure 2.5. The NLP parsing techniques lead to an indexed set of documents that are compared with a user query. After that, documents are scored and ranked by using machine learning techniques (MLR) and the results are displayed on a page. It is a complete view with functional flow of an IR search system, but it should be noted that document search only encompasses a part of the IR domain.

Information retrieval techniques consist of relatively simple tasks such as document searching and access, but also more complex tasks such as grouping of documents via clustering or classification techniques (Natarajan et al. 2005). An exemplary technique is the retrieval of relevant documents with respect to a user query using Boolean logic, where each document is represented as a vector of words. More advanced techniques use statistical methods in combination with machine learning to cluster documents (Hearst 1999) and these techniques have been getting more and more attention over the years (Cordón et al. 2003).

A specific branch of the IR domain is interactive information retrieval (IIR), which combines information retrieval with information behavior and human computer interaction domains and focusses more on the interaction of the user with the system (Kelly & Sugimoto 2013). It adds two elements on top of the IR process: pre-search techniques and post-search techniques. Pre-search techniques focus on assisting the user with the query and post-search techniques use the results of the system and relevant feedback to improve the information retrieval (Salton 1970).

2.3.3. Information Extraction (IE)

A second research domain that uses the NLP techniques heavily is the Information Extraction (IE) domain. Research in this domain aims at identifying pre-defined classifications such as events, locations, persons and represent that information in a structured format or template (Cowie et al. 1996). Where IR focusses on selecting relevant documents from a large set of documents, IE aims to identify entities in documents based on pre-determined classifications. This work dates back to the '70s where scripts/templates were designed for automatic classification of important entities from texts (e.g. Schank & Abelson 1977). Most of the progress in this domain however stems from the MUCs (Message Understanding Conferences) that ran from 1987-1997 to promote the design of IE systems, where IE systems could be discussed and evaluated (Small & Medsker 2014).

A distinction between two approaches can be made in IE systems: knowledge engineering approaches (or rule-based approaches) and machine learning approaches (Farley 2001). In the knowledge engineering approach, a developer of the system analyses text and writes rules for the extraction of information from the text. On the other hand, in the machine learning approach the developer annotates text and the system itself learns and creates rules on what to extract. Initially most systems relied on rule-based techniques, but more and more machine learning techniques are created (Small & Medsker 2014). This distinction is clearly made in the IE domain, but it also exists in the IR domain though less often expressed as such (Sebastiani 2002).

According to Appelt et al. (1995), a generic IE system consists of ten modules. First, a *text zoner* and *preprocessor* transforms the text in segments and a sequence of sentences. These sentences are then *filtered,* by eliminating irrelevant sentences. A *pre-parser* then tries to identify small structures in the text and subsequently a *parser* creates a parse tree fragments of the identified structures and the *fragment combiner* creates a full parse tree for the document. As a next step, a *semantic interpreter* generates semantic structures from the parse tree and a *lexical disambiguates* matches lexical information to the structures. The *coreference resolver* then identifies which entities in the tree represent the same meaning so that these are not mentioned multiple times. Finally, a *template generator* fills the IE templates from the semantic structures. Often multiple aspects of the above mentioned 10 modules are combined. As can be observed, an IE system relies heavily on the NLP domain and for this reason they are often mentioned together in literature (Cowie et al. 1996).

The Information Extraction modules show similarity with the Information Retrieval phases (e.g. indexing of the documents) and therefore IR is sometimes viewed as a first step for IE (Small & Medsker 2014). On the other hand, one can also argue that information extraction is part of information retrieval as IE processes can be used to identify entities in text that can be used for effective information retrieval based on these entities (Cowie et al. 1996). It can therefore be concluded that there is clear overlap between the IR and IE domain, as well as the techniques used from the NLP domain.

2.3.4. Text Mining (TM)

A third related domain is Text Mining (TM), which is part of the data mining domain that by itself is a branch of artificial intelligence (Liao et al. 2012). Text mining is defined as the discovery of “non-trivial, implicit, previously unknown, and potentially useful patterns” in text (Hearst 1999). Text mining dates back to the 1960s, but the techniques developed at that time were limited due to a lack of processing power of computers (Carbonell et al. 1983). The interest for this domain boosted again from 2000 onwards due to the exponential growth of processing power and unstructured information on the Internet, which requires techniques that are able to work with unstructured and unknown data and (Liao et al. 2012).

As demonstrated, Information Extraction methods can extract known information or entities from text, but they are not able to deal with unknown useful information in texts. Text mining bridges that gap as TM techniques do not use pre-determined features or classifications, but uses disciplines as artificial intelligence, statistics or machine learning to uncover patterns in structured data (Natarajan et al. 2005; Spasic et al. 2014). It can therefore be used in a complementary fashion to IE and it can be used to assist in IR methods. Text mining methods can establish connections between documents not known before and these connections can be used to determine relevant sets of documents to a user query (Martin 1995). In that regard, text mining methods do not use rule-based techniques but solely rely on advanced machine learning techniques (Sebastiani 2002).

The most common text mining task is document or text clustering and an exemplary functional flow diagram is depicted in Figure 2.6 (Gunjal 2014). Text clustering is a task where documents are grouped to clusters based on keywords identified by a machine learning algorithm. There is no human involvement in any of the steps, the keyword identification and similarity computation are performed by the algorithm. It is a very useful method to determine unknown but useful connections between documents (Jain 2010).

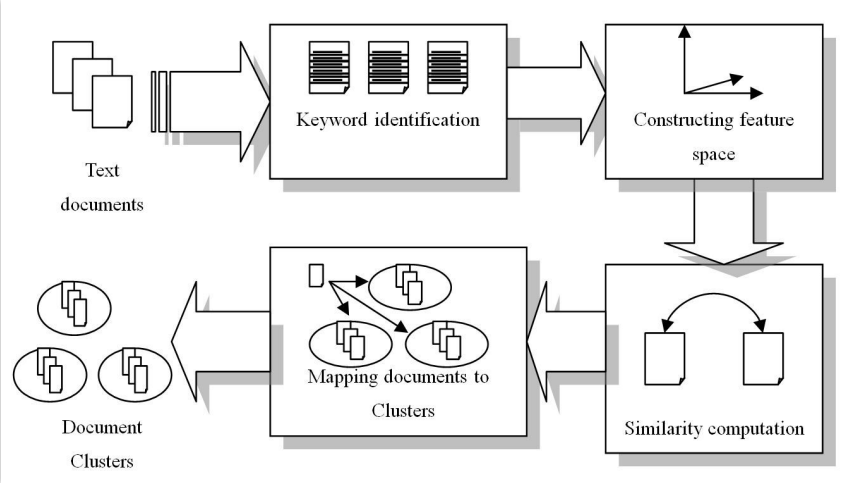


Figure 2.6: A text clustering framework (Gunjal 2014)

2.3.5. Machine Learning (ML)

So far the IR, IE and TM domains have been described that use NLP techniques to accomplish their goals. Most of the early work performed in these domains was based on rule-based or knowledge engineering techniques, that use manual rules created by an expert in the domain (Sebastiani 2002). A simple example would be an IF-THEN clause for information retrieval, IF the word exists in the document, THEN the document is relevant. As of the '80s, another set of techniques based on machine learning gained popularity (Cunningham et al. 1999).

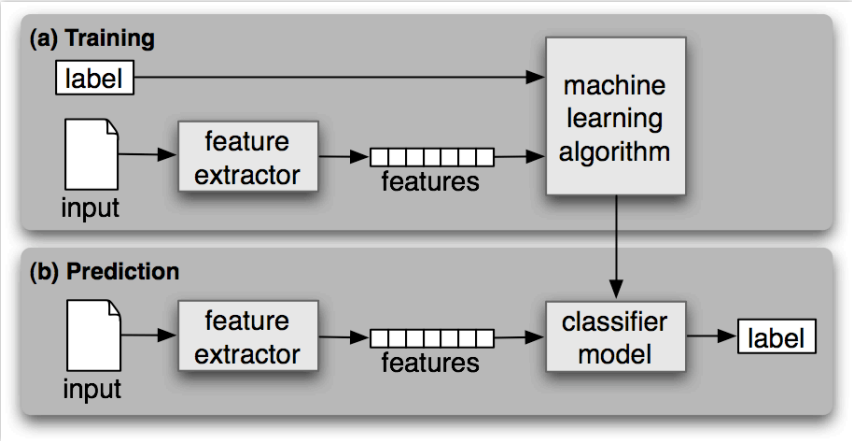


Figure 2.7: A supervised learning functional flow diagram (Bird et al. 2009)

Machine Learning (ML) is a broad field that covers supervised learning and unsupervised learning techniques. Supervised learning techniques are so-called "expert systems" that learn from a training dataset to create their own set of rules to predict what is requested by a user (Langley & Simon 1995). An exemplary functional flow diagram for a supervised learning system is shown in Figure 2.7. It can be seen that the system needs training data to function, which is based on manually annotated data (Bird et al. 2009). Text classification, as described in the IR domain as well, is the most common supervised learning task. Features (or classes) are chosen by the user and training

data is created, after which the system learns from this training data to predict the class of new documents.

On the other hand, unsupervised learning techniques – the “intelligent agents” - do not use training data and attempt to learn from documents by themselves and it is therefore a set of techniques closely related to the Text Mining domain (Hendler 1996). The text clustering framework shown in Figure 2.6 is an exemplary unsupervised learning method. Sometimes semi-supervised learning systems are mentioned in literature that use a limited amount of training data and annotate more training data themselves, but in essence these can be regarded as supervised systems (Small & Medsker 2014). Without going into detail of the techniques available in the machine learning paradigm, the IR, IE and TM systems mentioned in the previous sections all can benefit from learning systems versus the more conventional and simple rule-based systems.

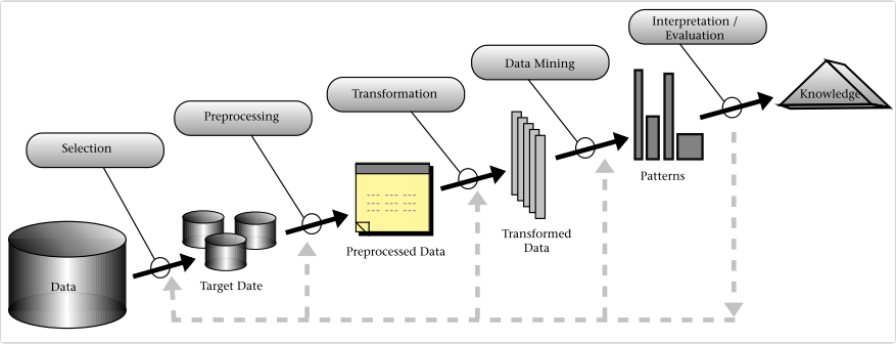


Figure 2.8: KDT process steps (Fayyad et al. 1996)

2.3.6. Knowledge Discovery in Text (KDT)

As the Information Retrieval, Information Extraction and Text Mining goals have some overlap, e.g. text mining and information extraction methods can be used for effective information retrieval, new governing research domains were developed that exploit this overlap. The first known research domain doing this is Knowledge Discovery in Databases (KDD), which originated in the 1990s. Knowledge Discovery is defined as “the nontrivial extraction of implicit, previously unknown, and potentially useful information from given data” (Frawley et al. 1992). Typical steps of the KDD process are visualized in Figure 2.8. Starting with the data, the steps selection, preprocessing, transformation, data mining and interpretation / evaluation can be observed before knowledge is obtained. Brusic & Zeleznikow (1999) and Small & Medsker (2014) add one similar first step to this process: identify the requirements of the user and the knowledge required for the domain of the data. This step can entail for example selecting an ontology or taxonomy relevant to the domain. It can be observed that data mining (or text mining) is an important part of the KDD process.

The research effort in the KDD domain is focused at knowledge discovery in structured databases, while most of the online information is in collections of unstructured text (Feldman et al. 1998). A new stream of research was established later in the 1990s with a focus on knowledge discovery in structured data but also on unstructured data: knowledge discovery in text (KDT). It is first mentioned by Feldman & Dagan (1995) and it is defined as “the study of techniques for dealing with unrestricted textual patterns and their evaluation for interestingness” (Perrin & Petry 2003). The goal of KDT is to understand patterns in both structured and unstructured text (Karanikas & Theodoulidis 2002).

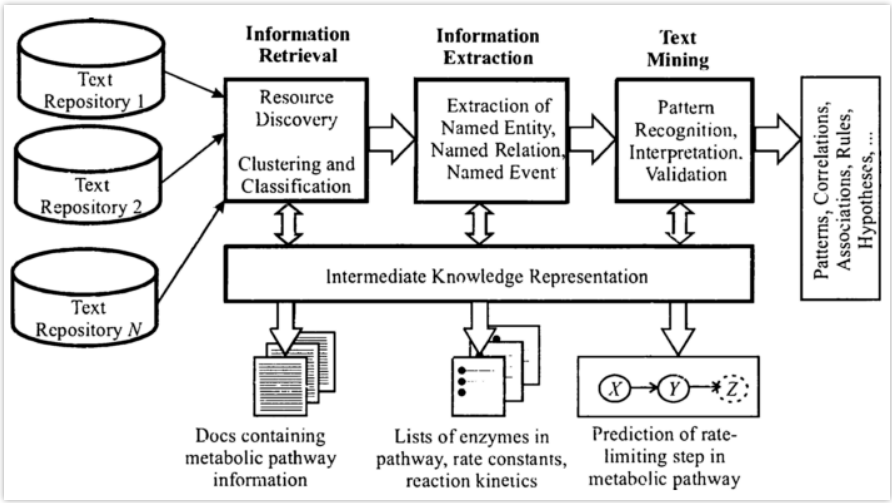


Figure 2.9: The KDT process as defined by Natarajan et al. (2005)

KDT consists of three previously mentioned domains: IR, IE, and TM. An exemplary relationship between those domains is visualized in Figure 2.9 (Natarajan et al. 2005). The three domains form the core elements for the KDT process and despite the fact that they are displayed in a sequential pattern in this visualization, in fact they have to be used in an iterative fashion until performance measures such as accuracy, time to completion or completeness are met (Perrin & Petry 2003). For instance, if an information retrieval process is not able to find enough relevant documents or sections based on a given query, then the information extraction or text mining step can be used to classify or cluster specific documents or sections in documents. This can allow the information retrieval step to perform better for the pre-defined query.

The IR techniques can be used to obtain a subset of relevant maintenance documents. This output can be further processed using the IE and TM techniques, but it can also be the final output: having a small set of documents from which required information can be extracted manually by the aircraft technician (Hakenberg et al. 2004).

2.3.7. Synthesis of the NLP, IR, IE, TM, ML and KDT domains

The research domains as described in the previous section can be summarized as follows:

- *NLP*: a set of techniques for pre-processing (parsing) of documents
- *IR*: set of methods for selecting a set of relevant documents
- *IE*: approaches for identifying known entities from text
- *TM*: methods for extracting unknown but relevant information from documents
- *ML*: using machine learning techniques for classifying or clustering of documents
- *KDT*: discovering (un)known relevant information from documents; IR+IE+TM

All given research domains provide some approaches or techniques that can be used to fulfil the research objective. Unfortunately, no comprehensive and complete overview is available in literature that covers all six research domains and their links / interfaces. The KDT process as visualized in Figure 2.9 covers most of the elements, but the NLP and ML domains are not adequately described in this overview. Additionally, the IR search system as depicted in Figure 2.5 demonstrated the links between NLP, IR, MLP and IE to some extent, but it leaves out the text mining steps. Therefore, a novel comprehensive overview is created in the form of a functional flow diagram (FFD) that covers the functionalities from all six domains adequately and it is shown in Figure 2.10.

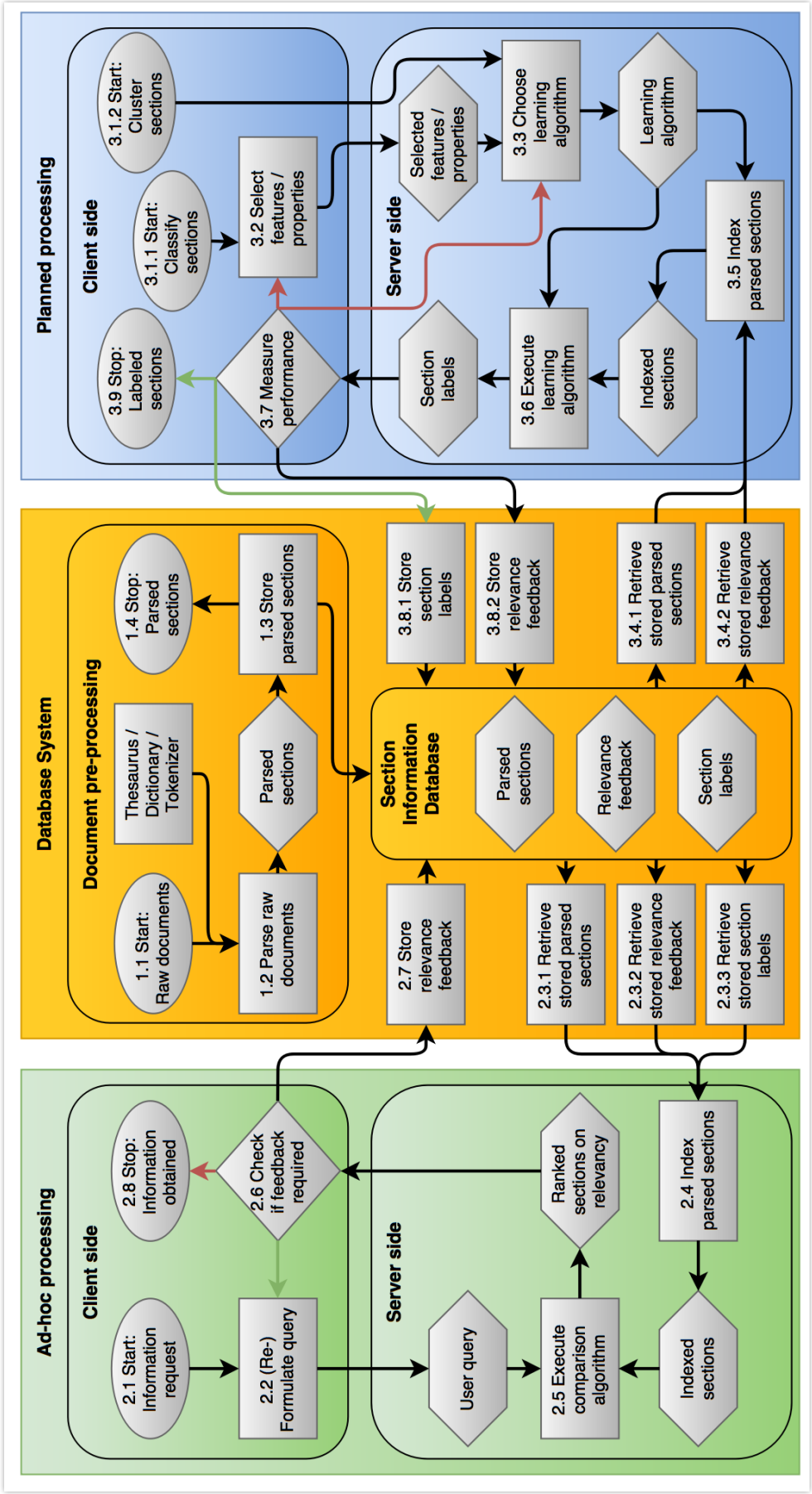


Figure 2.10: FFD with combined functionalities from all domains

The created FFD incorporates elements and functions from all research domains and it is created with the user of the system (the AMT) in mind. Three main elements of a system can be identified: a central database system, ad-hoc processing functions and planned processing functions. The central database consists of all functions related to the documentation themselves and the information related to the documents. The functions indicated with 1 are pre-processing techniques required to store the raw maintenance documents in a more accessible format for future reference by the ad-hoc and planned processing functions. Function 1.1 describes the start of the document pre-processing and it is immediately followed by function 1.2 that contains the parsing of the raw documents, supplemented by a thesaurus, dictionary or tokenizer. In other words, the parsing of documents means transforming from a pdf or full-text format to a parse tree per word or per sentence. A parse tree is very similar to a decision tree, but then it consists of all the words or sentences in a document (Cowie et al. 1996). Therefore, from function 1.3 onwards the documents are converted to sections that are defined as parsed representations of the documents. A section could be a full document or a chapter, paragraph or sentence. These sections are then stored in the Section Information Database and then this pre-processing of documents is finished with function 1.4. Most of the functions in this first part are based on core NLP and IE methods.

The second system element is ad-hoc processing, which is based on the IR systems described in the previous section. The functionality starts at 2.1 with an information request by the user of the system. A query is formulated based on the information need, e.g. relevant documents related to an aircraft type, error code or specific document. At the same time, three information sets are retrieved from the database: stored parsed sections (2.3.1), stored relevance feedback (2.3.2) and stored section labels (2.3.3). The stored parsed sections refer to the functionality described above in the first system element. Additionally, stored relevance feedback and section labels are generated by other parts of the system that is described later, but in short they indicate specific information about the sections. The stored sections, feedback and labels are used to perform indexing of 2.4, which is the representation of the sections in a useful format for comparison with the query. Afterwards, the sections and query are matched in function 2.5 and the sections are ranked based on their relevancy to the query. These are then send back to the user who determines whether this result is good or if the query must be reformulated (back to function 2.2). Additionally, in the same action relevance feedback is stored in the database (function 2.7). If the check for feedback results in no action required, the ad-hoc processing is completed (2.8). The functionality relies heavily on the IR domain, but also KDT, IE, NLP and ML functions are / can be used.

Lastly, the third element of the overview in Figure 2.10 is called planned processing. It indicates another form of pre-processing, this time not only on the documents but also on the metadata of the documents. Two major tasks for planned processing are text classification and text clustering. These are found in the IR, TM and ML domains. They are very similar in their functions, but a difference is found at the start of the functionality. After initiating classification (function 3.1.1) or clustering (function 3.1.2) the classification goes to selecting features whereas the clustering skips this step and immediately goes to choosing a learning algorithm (3.3). This difference in functionality stems from the way classification and clustering work: classification tries to group sections based on pre-determined classes or features and clustering aims to group sections based on unknown features of the documents. The feature selection function (3.2) for classification is performed manually. Afterwards an appropriate learning algorithm is chosen by the system (3.4). At the same time, stored parsed sections and relevance feedback are retrieved from the database (3.4.1 and 3.4.2 respectively) and indexed using the learning algorithm (3.5). Each learning algorithm uses its own representation of the sections, however a few representations are common for multiple algorithms (Small & Medsker 2014). Simultaneously, relevance feedback is added to the indexed sections for

the classification task, as these systems are supervised learning systems that work with training data (which is in fact the relevancy feedback on sections). This indexed representation of the sections is sent to the classification or clustering learning algorithm which then creates section labels (function 3.6). Subsequently, the performance is then measured in function 3.7; the methods for evaluating performance are described in the next section of this review. If the performance is below a threshold, the system either goes to reselecting features or chooses another learning algorithm. If it is above the threshold, the system stops (function 3.9) after storing the section labels and relevance feedback in the database (function 3.8.1 and 3.8.2).

Every system will need document pre-processing and ad-hoc processing functions to be able to get relevant information from documents based on an information request. The planned processing functions however are optional; these can be used to process expected information requests beforehand so that when the user needs it the relevant information can be provided immediately. While optional, these can be important when time is a critical factor in user information requests.

To sum up the linkage of the FFD with the six generic domains, first it should be noted that the document pre-processing system is mostly influenced by the NLP domain. IR functions are mostly visible in the ad-hoc processing side, but they are also used for text classification it can be viewed as a specific information retrieval task. However, the processes of text classification as well as text clustering are mostly influenced by the ML domain, as it has become the most dominant technique for text classification and clustering since the late 1990s. Furthermore, functions from the IE domain are not specifically mentioned in the flow diagram as this domain mostly describes tasks instead of functions. The functionality used in the IE domain is comparable to the IR domain. Additionally, the TM domain is also only partially present in the flow diagram, as it mostly covers text clustering and some document pre-processing tasks, but these functionalities are already clearly described in the ML and NLP domains respectively. Lastly, no specific functionality from the KDT domain is used, but the combination of IR+IE+TM domains in the KDT domain is extended by combining the IR+IE+TM+ML+NLP domains in this functional flow diagram. The KDT domain mostly describes generic overviews of how domains can be combined, but not yet how functionalities from domains can be combined in such a specific way. The novel FFD is not domain-specific and it can therefore be used as a generic model for any identification/extraction task. It can be applied in a domain-specific context, when a thesaurus or ontology specific for the aircraft maintenance domain is used.

With the generic research domains elaborated upon and summarized in the created FFD, the next sections focus on the functions used in this FFD. To be more specific, section 2.4 focusses on performance measures for classification and clustering systems as these performance measures can then be used to evaluate different techniques and methods available in literature. Afterwards, sections 2.5-2.8 elaborate on each element of the flow diagram, which are in order: document pre-processing techniques (section 2.5), ad-hoc techniques (section 2.6), classification techniques (section 2.7) and clustering techniques (section 2.8).

2.4. Assessment criteria for techniques

Before elaborating on the techniques to fulfil the functions depicted in the FFD, this section describes various measures to evaluate the performance these functions. All tasks of the functions to identify and extract information from documents can be viewed as a classification task, either information matches the information request (YES class) or not (NO class). Therefore, this section first describes classification assessment criteria in subsection 2.4.1, after which other assessment criteria are discussed in section 2.4.2. Lastly, a synthesis of assessment criteria is presented in section 2.4.3.

2.4.1. Classification assessment criteria

All classification assessment criteria are based on a confusion matrix, as is displayed in Table 2.1. For a given response of a classification function there are four options: a true positive (TP) indicates a correct positive prediction, a true negative (TN) indicates a correct negative prediction, a false positive (FP) is an incorrect positive prediction and a false negative (FN) is an incorrect negative prediction. In statistical terms, a FP is a type I error and a FN is a type II error.

Table 2.1: Confusion Matrix

	Actual positive	Actual negative
Predicted positive	True Positive (TP)	False Positive (FP)
Predicted negative	False Negative (FN)	True Negative (TN)

Based on these four options, multiple assessment criteria or metrics can be established (Davis & Goadrich 2006; Spasic et al. 2014) and these are described below in this section: recall, precision, specificity, negative predictive value, accuracy and F-score. These measures are explained in the context of a system that is used to classify documents based on relevance to a query.

Recall

Recall is defined as the ability to find positive matches and can be calculated by dividing the true positives by the true positives + false negatives. A high recall means that the system only misses few relevant documents that should be relevant to a query, i.e. many true positives and few false negatives. This is a useful measure if a system should not miss any relevant information to its user. The recall of a system is sometimes referred to as its sensitivity (Spasic et al. 2014).

$$R = Recall = \frac{TP}{TP + FN} \quad (0 < Recall \leq 1)$$

Precision

Another often used measure is precision, which can be calculated by dividing true positives by the true positives + false positives. In other words, precision is the amount of positive predictions that are correctly predicted. A high precision therefore means that most predicted relevant documents are relevant to the given query, i.e. many true positives and few false positives. Precision is a relevant measure if obtaining only relevant (and no irrelevant) information to the user is important (Davis & Goadrich 2006). Another word for the precision is positive predictive value.

$$P = Precision = \frac{TP}{TP + FP} \quad (0 < Precision \leq 1)$$

Specificity

A third less-used measure to assess classification systems is specificity (Spasic et al. 2014). It is determined by dividing the true negatives by the true negatives + false positives. It is a complementary measure to recall as it indicates the amount of correctly negative predicted relevant documents to a query. A high specificity indicates that most actual irrelevant documents are classified as irrelevant, e.g. many true negatives and few false positives.

$$Spec = Specificity = \frac{TN}{TN + FP} \quad (0 < Spec \leq 1)$$

Negative predictive value

A fourth measure that is also less used than precision and recall is the negative predictive value (Spasic et al. 2014). It can be calculated by dividing the true negatives by the true negatives + false negatives. It is a complementary measure to precision as it measures the amount of negative predictions that are correctly predicted. A high negative predictive value means many true negatives and few false negatives, which means that most irrelevant documents identified are irrelevant to the query. It is a useful measure if it is important to demonstrate that no relevant documents are missing from the response to the user.

$$NPV = \ Negative\ predictive\ value = \frac{TN}{TN + FN} \ \ (0 < NPV \leq 1)$$

Accuracy

The accuracy of a system can also be calculated using the above mentioned options and is used as a measure for the performance of classification systems (Spasic et al. 2014). It is calculated by dividing the true positives and negatives by all predictions (true positives + true negatives + false positives + false negatives) and therefore is a measure of how able the system is to predict actual relevant and irrelevant documents. A high accuracy indicates that the system makes few mistakes, i.e. false predictions.

$$Acc = \ Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \ \ (0 < Acc \leq 1)$$

F-score

A final measure in the category of classification assessment criteria is the F-score of a system, which is also the most often used measure to assess such systems (Spasic et al. 2014). The calculation can be seen below, the parameter β is a non-negative real number that can be used to determine the weight of precision relative to recall. For example, if precision is much more important than recall, β should be a high value and vice versa. The most common use however is $\beta = 1$, which means that precision and recall have the same weight (Wimalasuriya & Dou 2010; Ramakrishnan et al. 2012). Therefore, the F-score is sometimes mentioned as the F1-score, which refers to a β of 1.

$$F_\beta = \frac{(1 + \beta^2) * P * R}{(\beta^2 * P) + R} \ \ (0 < F_\beta \leq 1)$$

The F-score can be used on a two-class and multi-class set-up. The basic equation as depicted above works on two classes, as a classification outcome is either a correct or incorrect. The equation can be extended to work on multiple classes (3+) as well (Sokolova & Lapalme 2009). As only precision and recall are used for the calculation of the F-score, only the true positives, false positives and false negatives are required in each case. An example of results for a 4-class problem is given in Table 2.2. The values are randomly chosen just for explanatory purposes. A total of 570 results are divided in terms of predicted and actual class. The predicted classes are given in the rows and the actual classes for these results are given in the columns. For example, if a predicted result is class 2 and the actual class is 3, it should be in row 'Class 2' and column 'Class 3'.

Table 2.2: Multi-class F-score example - Input

		Actual			
		Class 1	Class 2	Class 3	Class 4
Predicted	Class 1	100	10	30	10
	Class 2	10	90	20	40
	Class 3	20	10	120	20
	Class 4	30	20	10	30

Table 2.3: Multi-class F-score example - Confusion Matrix

		Actual			
		Class 1	Class 2	Class 3	Class 4
Predicted	**Class 1**	TP	FP	FP	FP
	Class 2	FN			
	Class 3	FN			
	Class 4	FN			

The approach is comparable to the 2-class set-up, but to determine the F-score the individual scores for each class should be combined. This is demonstrated in Table 2.3, where the true positives, false positives and false negatives for the predicted value 'Class 1' is given. Using these values, the precision and recall for the predicted value 'Class 1' can be calculated. In a similar fashion, the precision and recall scores for the other classes can be calculated as is visualized in Table 2.4. Knowing all these sub-scores, they can be combined and divided by the number of classes to obtain the precision and recall for all results. The calculation of the final F-score is then following the $F_ß$ equation with ß = 1.

Table 2.4: Multi-class F-score example - Results

	Precision	**Recall**	**F-score**
Class 1	100 / (100 + 50) = 0.67	100 / (100 + 60) = 0.63	n/a
Class 2	90 / (90 + 70) = 0.56	90 / (90 + 40) = 0.69	n/a
Class 3	120 / (120 + 50) = 0.71	120 / (120 + 60) = 0.67	n/a
Class 4	30 / (30 + 60) = 0.33	30 / (30 + 70) = 0.30	n/a
Total	2.27 / 4 = 0.57	2.28 / 4 = 0.57	(2 * 0.57 * 0.57) / (0.57 + 0.57) = **0,57**

2.4.2. Other assessment criteria

Besides these classification assessment criteria, there are a few other assessment criteria that are used in literature. In the context of summarization of documentation Afantenos et al. (2005) mention a few qualitative assessment criteria for ordering of responses to a query: query-based ordering (which means ordering based on relevance to the query), salience-based (recitations, contradictions or other anomalies are weighted higher), domain-based (specific type of documents are weighted higher) and source-based (dependent relations between templates are displayed together). Additionally, Rajpathak (2013) demonstrates the use of rand statistics to measure class similarity with respect to class labels, which is a similar measure to accuracy of a system and therefore not further elaborated upon.

Furthermore, Batet et al. (2011) describe the use of other measures for evaluating systems, such as the computational complexity, dependency of knowledge sources and required parameter tuning. These measures focus more on the overall system than on the results of the system, but these are nevertheless relevant measures. Finally, Sebastiani (2002) provides a measure from the domain of decision theory for measuring the effectiveness of systems: its utility. In economics, the utility is defined as the satisfaction a consumer experiences from a product or services. It is yet another qualitative measure that could be used for evaluation.

2.4.3. Synthesis of assessment criteria

Almost all literature in the six research domains described in section 2.3 use three measures to assess a developed system or techniques: the precision, recall and F-score (Spasic et al. 2014; Small & Medsker 2014). Therefore, these measures are used to describe the performance of techniques or methods in the upcoming sections of this chapter. However, in the context of the system to be developed, other performance measures might be more relevant, such as the time required to perform a technique.

2.5. Document pre-processing techniques

This section describes the document pre-processing elements of the functional flow diagram. The related functions are all numbers with 1.X and highlighted in Figure 2.11. In brief, they describe the process of transforming the raw documents into sections that can be used in other parts of the identification and extraction process. The sections can be defined as either an entire document, but can also be a chapter, paragraph or even a specific word.

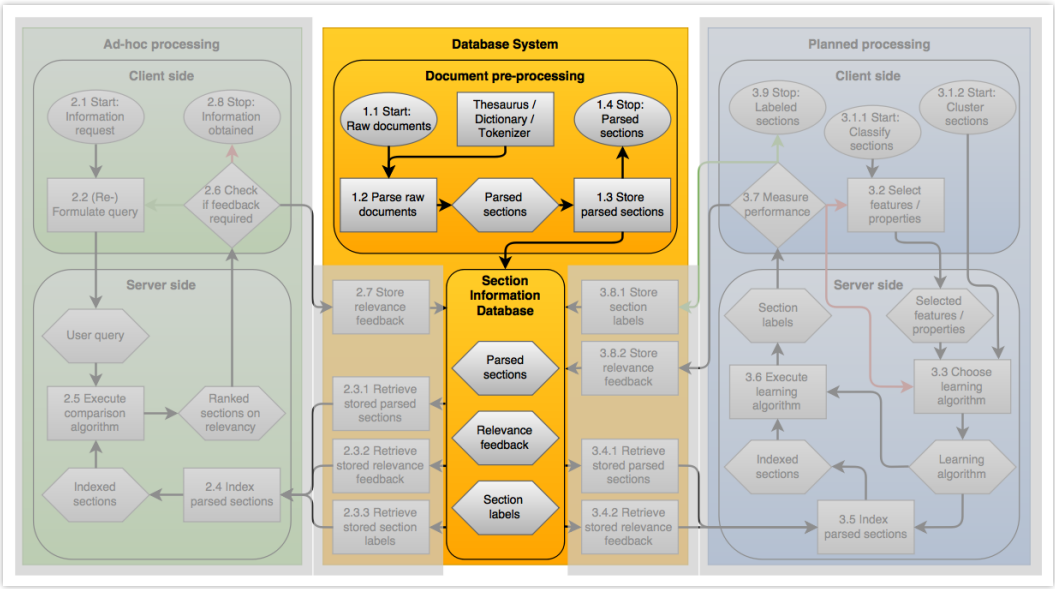


Figure 2.11: Document pre-processing elements highlighted in the FFD

PDF2Text

As a first step to process raw documents, text should be extracted from the raw documents. Maintenance documents are often in a PDF format and therefore should be converted to text that is accessible by more systems. There are many PDF to text conversion tools available and these are often even open-source. Examples of this software are LA-PDFText, Adobe Acrobat, IntraPDF, Grahl PDF Annotator, PDFTron and the most common one is PDF2Text (Ramakrishnan et al. 2012). These tools can break down texts despite layout difficulties such as tables and figures, and often represent the documents in a XML or HTML format. This format can much more easily be used by text processing tools. The obtained text sections can also be stored directly in a database system for easy access.

The Page Layout Recognition System (PLRS) as described by Esposito et al. (1995) is an example of the functionality, it can be viewed in Figure 2.12. Its functions are document scanning, deskew and complexity evaluation, reduction, page segmentation, layout analysis, document description, document classification, document understanding, Optical Character Recognition (OCR) and storing in a database. Interestingly, the functionality already uses a form of classification to determine the

document type so that it can use pre-determined rules to understand the document structure. The techniques required to identify the actual words in the document are described in the next subsection (mentioned as OCR in the PLRS system).

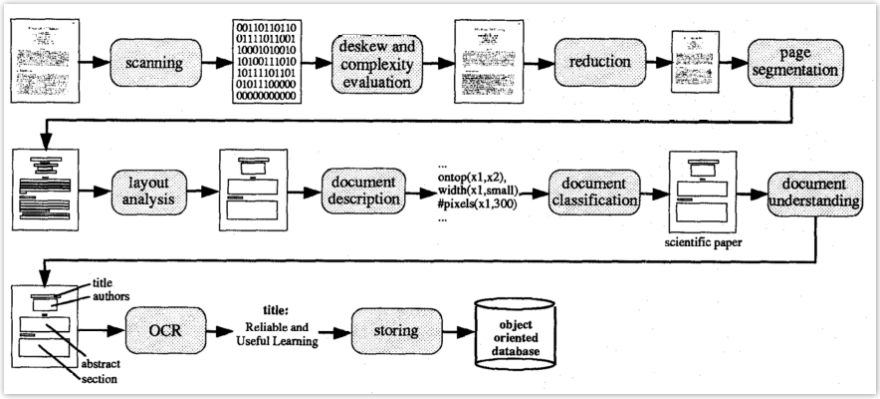


Figure 2.12: The Page Layout Recognition System by Esposito et al. (1995)

In case other document formats are used, such as XML, HTML or DOC files, much less elaborate techniques can be used for converting the text to a useful representation as these document formats already have the documents structure stored as metadata. Therefore, the techniques to convert these raw documents are not further described.

Text pre-processing

A specific step in the processing of PDF to text is the recognition of words in the text. The specific techniques required for recognizing letters in the text (OCR) are not elaborated upon as this is too detailed for the scope of this review. Instead, the NLP techniques that can be used to recognize words and sentences are described as these are one level higher and therefore useful for this study. Commonly used NLP techniques for text processing are the following (Dubitzky & Azuaje 2004; Collobert et al. 2011):

- *Sentence tokenization*: separating text into individual sentences (tokens).
- *Word tokenization*: breaking pieces of text into words (word tokens).
- *Part-of-speech tagging*: assigning part-of-speech information to tokens, e.g. article or noun
- *Stemming*: determining the stem of a word.
- *Shallow parsing*: identifying of elements of sentences (noun, noun phrase, etc.).
- *Full parsing*: constructing a complete parse tree (hierarchical structure) at sentence-level.

These described techniques are straightforward in their process, for example for sentence tokenization one task is to register dots in sentences and identifying which dots represent the end of sentences. For any task, often a combination of these techniques is used (Small & Medsker 2014).

Thesaurus / dictionary

Lastly, the use of a thesaurus or dictionary can be used as an addition to the text pre-processing techniques. For example, WordNet is developed by Princeton as a large lexical database that describes e.g. nouns, verbs and adjectives (Miller 1995). It consists not only of the part-of-speech information of the words, but it also has information on synonyms and synsets of the words, e.g. rifle and machine gun are classified as firearm, piece, small-arm; gun; weapon, arm, weapon system. In this way, another level of information is stored besides just the part-of-speech of the words. This information can be used for either the described basic pre-processing tasks, but they can also aid in a later stage with document classification techniques, e.g. in a task to recognize the similarity between documents.

Besides the use of a thesaurus or dictionary, also a (domain-specific) ontology can be used to aid the document pre-processing task. By providing for example common words and terms in the aircraft maintenance domain, the system is more effectively able to recognize part-of-speech information. In other words, ontologies can be used as an effective knowledge base based on domain-specific information to aid in identification and extraction of relevant information in text (Fernández et al. 2011).

Conclusion

This section described available techniques and tools in literature for pre-processing of raw maintenance documents, so that they can be stored in an accessible format in a database in relevant sections. It can be chosen to store the information at word-level, sentence-level, paragraph-level, chapter-level or document-level. For this review, the format in which the text is stored is called at section-level, which can encompass any of these formats.

2.6. Ad-hoc processing techniques

This section describes the available techniques in literature that can be used to fulfil the ad-hoc processing functions of the FFD, highlighted in Figure 2.13. The specific functions are formulating a query, indexing the parsed sections in an appropriate format for comparison of the query with the sections and finally a check if feedback is required. The ad-hoc functions are comparable to a typical IR system and therefore five major IR models can be identified that can fulfil the functions: Boolean models, vector space models, fuzzy models, latent semantic models and probabilistic models. Each of these models is described as well as how they fit in the functions of the ad-hoc processing part of the FFD.

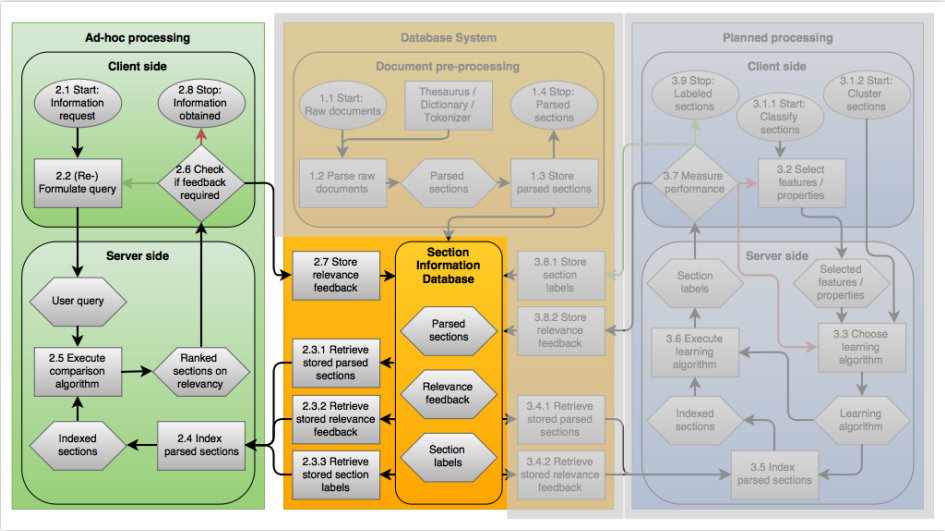


Figure 2.13: Ad-hoc processing elements highlighted in the FFD

Boolean models

As with any of the IR models, the goal of the functions is to rank the parsed sections based on relevancy to a query (Small & Medsker 2014). This can be performed by creating an index of the parsed sections on word-level and comparing this index with the query. One of the oldest models available is the Boolean model, which is developed from the start of the IR domain the 1960s (van Rijsbergen 1986). In a Boolean model a simple way of indexing is performed by creating a matrix of terms versus the sections where it is indicated whether a term is in a document or not, the Boolean value YES or NO (Manning et al. 2009).

Table 2.5: Example of Boolean model

	s_1	s_2	s_3	s_n		q_1	q_2
t_1	YES	YES	NO	…		t_1 AND t_2 → s_2	t_1 OR t_2 → s_1 and s_2
t_2	NO	YES	NO	…			
t_3	NO	NO	YES	…			
t_n	…	…	…	…			

An example is given in Table 2.5 where terms are the rows of the index and sections the columns, with a simple YES or NO indicating that a term exists in the system. Two simple query examples are given, query 1 indicates that sections with the term 1 AND term 2 should be returned which results in section 2. In a similar fashion, query 2 requests sections with term 1 OR term 2 and therefore both section 1 and 2 are returned. For such a model, simple operators such as AND, OR and NOT can be used.

Despite its simplicity, this model is still often used (Natarajan et al. 2005). However, the Boolean model is unable to rank sections based on relevancy, it is only able to return a set of sections that either match or do not match the query. Additionally, the Boolean model is not able to capture any semantic relationship between the terms in a query, e.g. if a search is performed for “aircraft maintenance” it is not able to understand the meaning of aircraft maintenance with respect to aircraft and maintenance. This problem cannot be solved with the simple Boolean model and a more advanced method has to be used that will be explained later in this chapter (see subsection ‘Latent semantic models’).

Furthermore, if terms are not represented as words but as classes, the Boolean model can be used effectively. For instance, if sections are classified based on the class Airbus (YES or NO), then a Boolean search for Airbus can simply return all sections that have that class Airbus. Therefore, a combination between a good classification system of sections and a Boolean search can already provide useful results. Nevertheless, the lack of ranking of results is a major drawback and therefore more advanced methods are often applied, such as vector space models.

Vector space models

To deal with the issue of ranking based on relevancy in the Boolean model, the vector space model was developed as a more advanced method (Salton & Buckley 1988). The main difference between the vector space and Boolean model is that sections are indexed using weighted terms. The weight of the terms can be determined by counting the frequency of the terms in a section or via a more advanced by multiplying the term frequency (tf) with the inverse document frequency (Manning et al. 2009). The inverse document frequency (idf) indicates the frequency of sections that contain the term and is calculated by taking the logarithm of: (total number of sections) / (number of sections containing the term). The logarithm is used to amplify the effects of uncommon terms that only appear in few sections. The term frequency thereby indicates the importance of a section (sensitivity) and the inverse document frequency captures the specificity of the term.

The vector space model compares the section vectors with a query vector containing certain terms that can be weighted if required. There are two common methods for determining the similarity between the sections and query, measuring the Euclidean distance or measuring the cosine coefficient of the vectors (Natarajan et al. 2005). The Euclidean distance between the section and

query vectors works for small vectors, but for larger vectors this distance measure can be less accurate. The cosine coefficient between the vectors is more applicable as vector length is not an issue. It is determined by calculating the angle between the query and section vectors and the size of the angle is a measure for the relevance of a document. The calculation for measuring the angle between the query vector and one section can be seen below, with q = query vector consisting of terms q_i; s = section vector consisting of terms s_i; N = total terms that exist in each of the vectors.

$$angle(q,s) = \frac{q \cdot s}{|q| \cdot |s|} = \frac{\sum_{i=1}^{N}(q_i \cdot s_i)}{\sqrt{\sum_{i=1}^{N} q_i^2 \cdot \sum_{i=1}^{N} s_i^2}}$$

If the angle between a section and query vector is small, their terms are similar and thus relevant. A cosine coefficient of 1 indicates two vectors at the same path (angle = 0°), a coefficient of 0 indicates an angle of 90° and -1 indicates an angle of 180°. An example is provided in Table 2.6, showing the term and section matrix and a query with all term frequencies randomly determined. For sections with an exact match with a query term the coefficient is much closer to 1 compared to terms that match the query (as for section 2). The amount of sections and terms are not representative for real situations are merely an illustration of the functionality and therefore no further conclusions can be attached to the numbers.

The vector space model is an effective method for word-based searches or class-based searched as explained for the Boolean model as well and it can weigh terms in the index both for sections and the query. In that regard, it is an often used model for syntactic searches (Cordón et al. 2003).

*Table 2.6: Example of vector space model using tf*idf weighing*

	s_1	**s_2**	**s_3**	**s_n**		**q_1**
t_1	3*0.176 = 0.528	2*0.176 = 0.352	0	…		0.5
t_2	0	1*0.477 = 0.477	0	…		0
t_3	0	0	2*0.477 = 0.954	…		0.5
t_n	…	…	…	…		…
angle(q,s) [rad]	0.264/0.373 = 0.708	0.176/0.506 = 0.348	0.477/0.675 = 0.707			

Fuzzy model

Another way to be able to rank documents but still use Boolean operations, is by using a fuzzy model (Cordón et al. 2003). Where a Boolean model uses either 0 and 1 for values, the fuzzy model defines a value between 0 and 1 and it is therefore regarded as the extended Boolean model. In a fuzzy model, the terms in the sections are weighted based on their relevance to the section. This is a different approach to the vector space model that uses term frequency as a weighing function. The fuzzy model compares the section term vectors with a query vector that can be setup using AND and OR clauses. The calculations are comparable to the vector space model and therefore not shown again for this model.

The fuzzy model is especially useful if the relevance of the content of sections should be considered and compared to the standard Boolean model it can use document weighing. It is however also more complex to use this method and it is only preferred over the vector space model if section contents are important to take into account (Fukumoto et al. 1997).

Latent semantic model

Where all three models above are used for syntactic searches, the latent semantic model extends that paradigm with searches based on semantics of terms (Deerwester et al. 1990). With the latent semantic model, semantic dependencies between terms can be considered. This is achieved by starting with the term versus section matrix and performing Singular Value Decomposition (SVD) on it, see Figure 2.14 for an example. The SVD creates three matrixes of the original one, where especially the m x m matrix is relevant for the process. This matrix consists of singular values of the original term versus section matrix, which are a set of concepts that can be found in the terms and sections. Without going too much into detail in the algebraic operation, the singular values or concepts are matched with the terms and sections in the other three matrices (t x m; m x m; m x s). When a search is performed on either one of the terms or concepts, the singular value matrices can match the term to the concepts (t x m matrix) and the terms or concepts to the sections (via the m x m matrix or the m x s matrix).

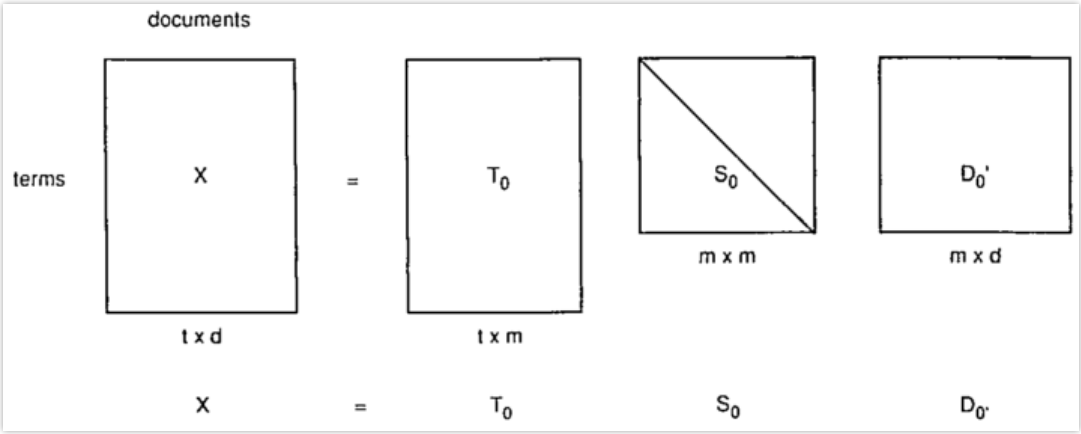


Figure 2.14: Latent semantic model; SVD example (Deerwester et al. 1990)

The latent semantic model is therefore useful for two reasons. First of all, by decomposing the original term x sections database the amount of information is often greatly reduced, because the original terms versus sections databases mainly consists of zeroes as most terms are not found in all sections (Sebastiani 2002). Secondly, it allows to search at a semantic level, which for some contexts is much more relevant than a syntactic search, especially if the information required is not clear at the start of the information retrieval process. This way of indexing the terms and sections can help to find related sections that do not have the syntactic word that is written in the query (Fernández et al. 2011). On the other hand, if a search term is very good in describing the documents or sections required, this method can provide less relevant results by using the semantic concepts related to it (Natarajan et al. 2005). It is both an advantage and disadvantage and therefore depends on the task whether it should be applied.

Probabilistic model

So far all models described did not use any relevance feedback (or learning) to improve the search process, they all rely on the quality of the indexing method and the query. If a user is unable to formulate a well-defined query for his information need, these models can lead to poor results. A last

model in this chapter that does use the relevance feedback to improve information retrieval is the probabilistic model. The probabilistic model stems from the core of the problem: both the query as the section representation lead to uncertainty with respect to the information need and probabilistic methods are founded on these uncertainties (Manning et al. 2009).

The most common used probabilistic model is the Binary Independence Model (BIM) that uses a similar approach as the simple Boolean model (Sebastiani 2002). Terms are represented as Boolean values (the Binary part of BIM) with respect to sections and queries. The Independence part refers to the fact that terms are modelled independently to the documents, so no dependency between terms and sections is assumed in this model. This is not a valid assumption, but nevertheless very good results are achieved with this model (Manning et al. 2009). The mathematics behind the probabilistic model is too extended to elaborate upon, but in sum probabilities are calculated for the relevance of query terms in the sections. These probabilities are based on initial estimates if there is no relevance feedback yet, and after relevance feedback is obtained from the user (for example is he checks the results and decides to alter the search query) they can be better estimated.

This model is especially interesting if relevance feedback is available so that accurate probability estimates can be calculated for retrieving relevant sections. In combination with classification techniques that require training data (which in fact relies on relevance feedback) these can create a good model for information retrieval. The next section of this chapter focusses on these classification techniques.

2.7. Planned processing techniques: Text Classification

This section describes one of the planned processing set of techniques: text classification. Text classification deals with the assignment of classes or labels to sections and often based on a supervised machine learning algorithm. It can be performed manually as well, but there are no techniques to explain for that process. The text classification functions perform similar steps as the ad-hoc processing functions, because text classification can be regarded as a form of information retrieval (and thus ad-hoc processing). The classification starts with selecting features (or properties) that should be assigned to sections, as can be observed in Figure 2.15. Subsequently, a learning algorithm is chosen that fits with the specified features. At the same time, stored parsed sections and relevance feedback are retrieved from the database as these form the data input for the task.

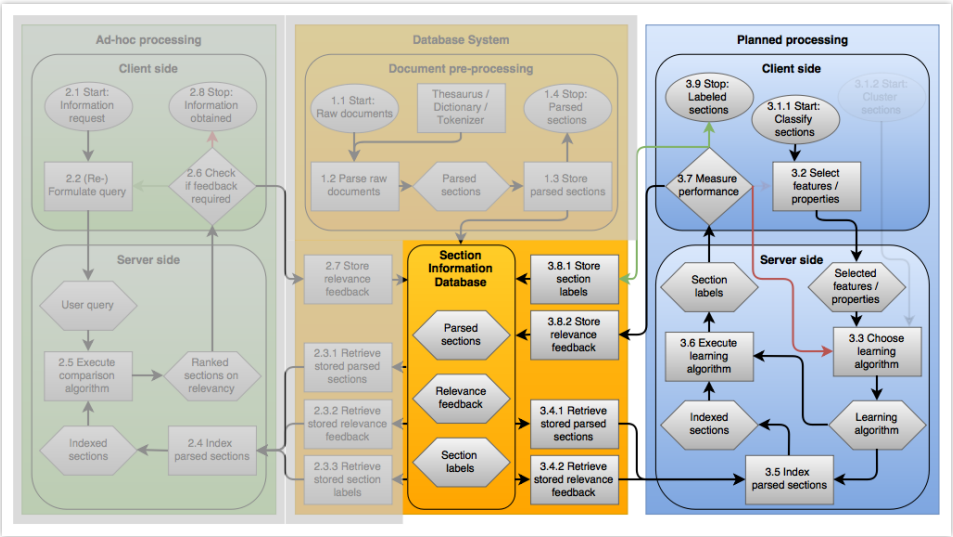


Figure 2.15: Classification functions highlighted in the FFD

The relevance feedback can be regarded as training data for the learning algorithm; the training data is a set of known predetermined sections with the right class and the learning algorithm will apply this to the rest of the sections. Based on the learning algorithm, the parsed sections and relevance feedback, the sections are indexed accordingly and then the learning algorithm is executed. The result is a set of section labels that can be evaluated using performance measurements, for example the precision, recall and F-score. If the performance is above a certain threshold, the section labels are stored in the database, otherwise different features and/or another learning algorithm is chosen to redo the process.

The two following sections discusses both the classification tasks (2.7.1) that can be performed and the techniques (2.7.2) that can fulfil the functions. A selection for most common both tasks and techniques is made as there is an abundance available in literature.

2.7.1. Classification tasks

There are two major ways to perform the classification task, either class-based or document-based (Sebastiani 2002). Class-based classification is performed when a new class is added or a class is changed, while document-based classification is performed when a new document is added to the document set. It is a pragmatic difference rather than a conceptual one, but it is important to distinguish between the two approaches. For a class-based classification the learning process must be initiated from scratch and is executed for all documents, while document-based classification builds on the learnings from previous documents. For the task at hand both document- and class-based classification will be used. Document-based classification will be used as there will be an initial set of documents and new documentation will be added (for example Service Bulletins). Class-based classification is used when new classes are added, e.g. if a new aircraft type is added and documentation should be classified whether it is related to this new aircraft type.

Additionally, the classification can be performed based on hard versus ranking classification (Sebastiani 2002). Hard classification refers to cutting off specific sections that do not match the determined feature, for example not adding the label to a section that has Boolean 0 match with a feature. Ranking classification adds the class to every document only then with a specific weight attached to it, e.g. a document can be given the label “A330” with weight 0.2. Making this distinction in classification type helps to filter specific documents below a certain threshold, for example.

NLP classification tasks

Classification can be performed by a user request to classify based on a specific word (the word is then the feature), but it can also be performed on more generic features in the text such as a named entity (a location, a name, etc.). These tasks stem from the Information Extraction domain and on the 7th Message Understanding Conference (MUC-7) a list of 5 tasks was identified (Kaufman 1998):

- *Named Entity Recognition (NER)*: e.g. recognizing ‘Delft’ as a LOCATION.
- *Co-reference resolution*: e.g. identifying ‘TU Delft’ and ‘University of Technology Delft’ as the same entity. But also, recognizing: the company fired 20 employees and they (=COMPANY) are not bankrupt.
- *Template element filling*: e.g. recognizing locations in organizations (Delft in TU Delft).
- *Template relation filling*: e.g. TU Delft is an organization but also a university (TU) and a location (Delft).
- *Scenario template filling*: e.g. He <person> works for the TU <org.> Delft <location>.

Additionally, Collobert et al. (2011) determined four standard NLP tasks that are used to classify specific information in texts:

- *Named Entity Recognition (NER)*: this task is also mentioned in the MUC-7 list above. It is the labelling of elements in a text such as a PERSON or LOCATION. A typical F1 performance, which is a combined measure of precision and recall, is achieved of around 85% (Ando & Tong 2005).
- *Part-of-speech tagging (POS)*: it aims at assigning part-of-speech information, labels and other markers to tokens, e.g. adjective, article or noun. It is a relatively easy task to perform and this is confirmed by average F-scores as high as 95-98% (Toutanova et al. 2003).
- *Chunking*: chunking is also called shallow parsing and it aims at identifying of elements of sentences (noun, noun phrase, etc.). It can be used either in document pre-processing, but also as a classification task. Typical scores for chunking tasks are around 90-95% (Collobert et al. 2011).
- *Semantic Role Labelling (SRL)*: it aims at providing semantic information to syntactic parts of a sentence, e.g. in the sentence 'He sold the book to him' recognizing who is meant with 'he' and 'him' based on previous sentences. Typical F1 scores for this task are between 75-80% (Pradhan et al. 2004).

A prerequisite for most of these NLP tasks is that they require a significant amount of training data to perform accurately (Small & Medsker 2014); scores mentioned previously can only be obtained if a sufficient amount of training is available, which requires a substantial time investment. However, if training data is picked from for example an existing database (e.g. WordNet) then the time investment is significantly less. It is an important issue to consider when synthesizing the classification techniques with other available techniques.

2.7.2. Classification techniques

To perform any of the classification tasks as described in the previous section or the classification on a chosen word, there are 2 distinct groups of techniques that can be applied: statistical and vector space techniques. The statistical techniques determine probabilities based on the training data to estimate the class of new documents (Nadkarni et al. 2011). Relevant techniques for text classification are the Naive Bayes technique, hidden Markov model and decision tree technique. Additionally, vector space techniques use distances between points in the vector space to determine the class of new documents based on training data (Small & Medsker 2014). Relevant vector space techniques for the task at hand are Support Vector Machines or instance-based methods, such as the k-Nearest Neighbors (kNN) method. There are many more classification techniques available in literature, such as regression and neural network methods, but these are less relevant for text classification purposes due to complexity or poor performance when working with text (Sebastiani 2002). The rest of this section gives an overview of the mentioned relevant techniques.

Statistical: Naive Bayes

A Bayesian approach uses a probabilistic method to text classification. In this approach, a probabilistic likelihood is calculated that a document matches a class based on training documents with known classes (Manning et al. 2009). For example, a test section should be assigned the class 'YES' or 'NO' based on whether it belongs to Airbus. Using a set of training sections with their class known, the probabilities are calculated for each word in the test section that it belongs to either class. The highest probability means the highest likelihood the test section matches therefore should be assigned. In Table 2.7 a numerical example of document indexing for the Bayesian approach is

given, with the frequency of chosen terms indexed for all documents, as well as the class for these documents. Furthermore, Table 2.8 shows the probabilities for the class of each term, calculated with the following equation:

$$P_{term}(YES/NO) = \frac{n_{term} + 1}{n_{total} + n_{voc}}$$

with: n_{term} = frequency of the term in case of YES or NO class,
n_{total} = total number of terms associated with YES or NO class,
n_{voc} = total number of unique terms or in other words the vocabulary.

Table 2.7: Bayesian example - Documents index

Terms:	*A340*	*787*	*A330*	*747*	*Flight*	*Wing*	*Manual*	**Class**
d_1	3		2		2	2		YES
d_2		4		5	7	2		NO
d_3	5		4		4		1	YES
d_4	3	1			2	4	4	YES
d_5		2			3		5	NO
d_{new}	2						2	?

Table 2.8: Bayesian example - Probabilities

Class:	*A340*	*787*	*A330*	*747*	*Flight*	*Wing*	*Manual*	**Class**
YES	(11+1)/ (37+7)	(1+1)/ (37+7)	(6+1)/ (37+7)	(0+1)/ (37+7)	(8+1)/ (37+7)	(6+1)/ (37+7)	(5+1)/ (37+7)	3/5
NO	(0+1)/ (28+7)	(6+1)/ (28+7)	(0+1)/ (28+7)	(5+1)/ (28+7)	(10+1)/ (28+7)	(2+1)/ (28+7)	(5+1)/ (28+7)	2/5

Now that the probabilities for each of the terms are known, the probability that the new document belongs to class YES and the probability that it belongs to class NO can be calculated by looking at the terms in the new document. For the example, the results can be seen in Table 2.9, with a clear higher probability for the class YES for the new document. Based on the Naive Bayes method, it can be concluded that the new document should belong to class YES.

Table 2.9: Bayesian example - New document probabilities

Terms:	*A340*	*787*	…	*Manual*	**Class**	**P(Y/N)**
YES	2*(11+1)/(37+7)			2*(5+1)/(37+7)	3/5	= 0.089
NO	2*(0+1)/(28+7)			2*(5+1)/(28+7)	2/5	= 0.008

Many extensions on the example given above can be done, e.g. using more than 1 class, using fuzzy logic for the class calculation or using all words in sections for indexing (Sebastiani 2002). It is a very simple model and because of that it is often used. Disadvantages of this model are that the results are highly dependent on the choice of indexed terms and therefore the right selection should be made. Also, this way of calculating the probability assumes the independence of all terms in each section (the naive assumption of this approach), which is often not valid (Small & Medsker 2014). However, despite this assumption very good results can still be obtained and therefore it is a common method for text classification.

Statistical: hidden Markov models

Where the Naive Bayes method assumed no dependence of the individual terms in sections, a method that is based on the dependence of terms in sections is based on a hidden Markov model (HMM). A hidden Markov model is a system where a variable can change between different states depending on observations and conditional probabilities (Frasconi et al. 2002). The 'hidden' part in the HMM name refers to the fact that the state (category) of a section is unknown and cannot be seen, but can be determined using a probabilities model. The main difference with the Naive Bayes technique is that in a HMM the category (or state) a new section belongs to depends on the sequence of terms in the sections. The specific algorithm that is applied is the Viterbi algorithm, which is used to find the most likely sequence of states based on observations (Nadkarni et al. 2011).

The functionality of the HMM using the Viterbi algorithm is best explained with an example and the one given in the Naive Bayes method can be used for this. In this context, the example can be stated as finding the class 'YES' or 'NO' for a new section based on a set of training data sections with predefined class 'YES' or 'NO'. Determining the class for the new section is based on a sequential method, each new term in the section is an observation at each observation the probability is calculated that the new section belongs to either class 'YES' or 'NO'. As mentioned, the main difference is that the HMM model assumed (sequential) dependence between the terms. Normally it would be used with full sentences instead of specific terms as in this example, but the functionality is the same so it can be used for explanatory purposes. Three probabilities should be calculated based on training data:

- *Original probabilities*: the probability that a new section belongs to the class 'YES' or 'NO', which are calculated to be 0.6 and 0.4 respectively for the given example.
- *Transition probabilities*: this probability indicates the likelihood that the state changes from observation to observation, e.g. the probabilities that the state changes from 'YES' to 'NO' and vice versa based on a given term. The computations are quite extensive; therefore, it is now assumed for this example that the probability that a class changes is always 0.5: Yes->Yes (0.5); Yes->No (0.5); No->Yes (0.5); No->No (0.5). This means that the interdependence of terms is zero.
- *Emission probabilities*: this is the probability that a given term belongs to a specific class, which can be calculated by the Bayesian probabilities as given in Table 2.8.

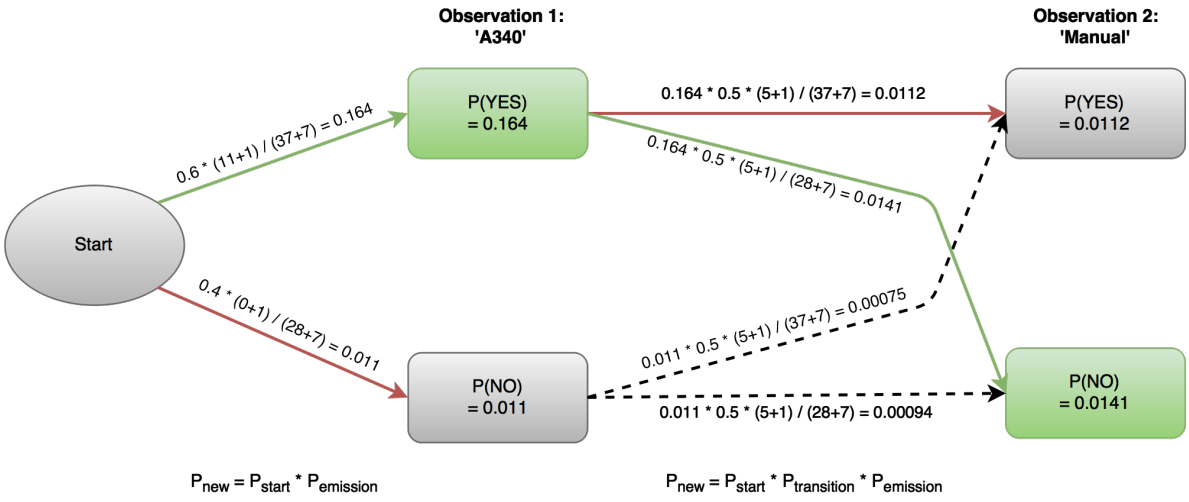


Figure 2.16: Calculation of probabilities in a HMM example

The probability for a new section can be calculated with these probabilities, see Figure 2.16. At the first observation, the term ‘A340’, the probabilities are calculated for the case ‘YES’ and ‘NO by multiplying the initial probability for the class YES and NO with the emission probability for A340 in case of YES and NO (see Table 2.8 for the emission probabilities). Based on only observation term A340, the section should be classified as ‘YES’ as the probability is higher. Subsequently, another observation is added: the term ‘manual’. Now 4 probabilities must be calculated as the transition probabilities are included from YES to NO and vice versa. A selection should be made which two probabilities are higher: the ones coming from P(YES) or from P(NO), which is this case are the ones from P(YES). Then these probabilities are used for the probability for class YES or NO after observation 1 and 2. It is found that now P(NO) has a higher probability and based on this sequence of observations the new section should be classified as NO. Of course, this is a simple example and based on only 2 observations with random terms and the answer is incorrect. It demonstrates the importance of choosing the right terms for this method. Additionally, it can also be observed that the term A340 indicates the class YES and in this case the class manual causes a shift to NO. This method can therefore be efficiently used to determine terms that are pivotal for determining the class of sections.

This method provides additional useful information compared to the Naive Bayes technique: transition probabilities indicate the independence of terms (if they are 0.5 then there is no independence, but if they are 0.8 and 0.2 respectively then it is clear that terms are more fixed to one class). In that regard, the hidden Markov model in this format adds another layer to the Naive Bayes method that can be quite useful.

Statistical: Decision tree

Where the Naive Bayes and HMM methods use probabilities based on terms in the sections and represents the probabilities in a large index, the decision tree uses a similar approach but uses a different way of representing the probabilities (Manning & Schütze 1999). In a typical decision tree, each node represents an important attribute related to the sections, each branch from that node is a possible value for that attribute and the final leaf of the tree is the category of the document. An example of this process is given in Figure 2.17 for the classification ‘should I play tennis?’ (Mitchell & others 1997). The attributes are outlook, humidity and wind and each attribute has its own values. The decision is made by moving in the tree top-down, and if the leaves of the tree are reached either the classification YES / NO is reached. In this simple example the values are also simplified (e.g. high / normal humidity) and often this binary way for building the tree is used, but it can be often extended with weighing based on probabilities (Sebastiani 2002).

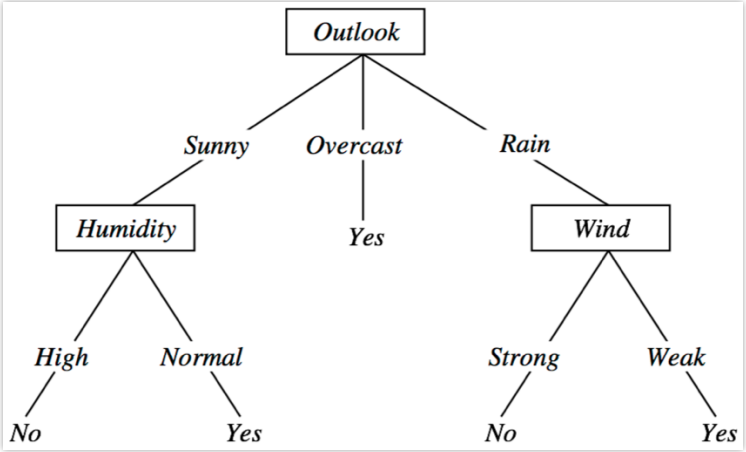


Figure 2.17: Decision tree for the task: should I play tennis? (Mitchell & others 1997)

The tree given in the example is created manually, but in a learning algorithm a tree is induced automatically from the training data using two phases (Uğuz 2011). In the first phase the tree is grown based on the contents of sections. The algorithm learns based on the contents of the sections what should be part of the tree and how it in the end results in the given classifications. The second stage of the algorithm is removing the branches of the tree that are not useful, as the algorithm easily overfits the training data (Fuhr & Buckley 1991). The decision tree is a useful method as it can be automatically induced and is very complete, however it can also become too complete (overfitting of the data) that results in performance loss. Therefore, the second stage of the process is important to cope with this.

Vector space: k-Nearest Neighbors (kNN)

The previous methods all focused on statistical calculations, but there is another set of classification algorithms based on the vector space of terms. The vector space methods work in a similar fashion as the vector space model discussed in section 2.6, but in this case all sections are plotted as a point in a x-dimensional space instead of a vector (Manning et al. 2009). The kNN algorithm is relatively straightforward, training data is plotted in this vector space and then a new section is added as a dot after which the 'k' closest dots are evaluated based on their class. The class that is most often present is the class for the new section. The best applicable cases are for 2 classes where it is recommended to choose k=uneven to avoid any ties. Multiple classes are possible, but then an additional measure should be applied such as weighing the distance from one dot to another to avoid any ties. It is technically possible to still get another tie, in which case a random pick has to be performed (Sebastiani 2002).

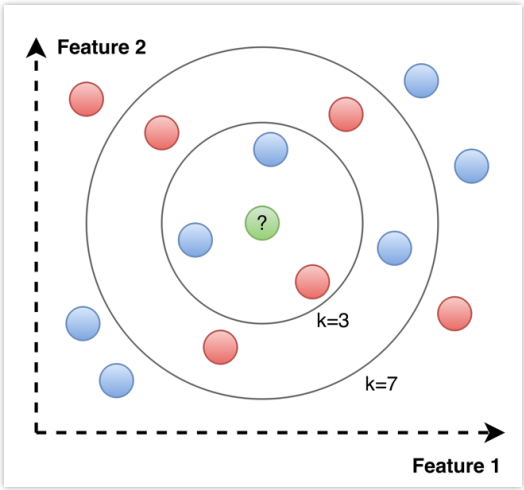


Figure 2.18: kNN example

A simple example of kNN is provided in Figure 2.18, where the k-nearest neighbors are selected for the green dot. In case of k=3, the blue category should be given to the unknown dot, but in case of k=7 the red category should be given. The selection of 'k' is very important and depends on the input data, for example the features that represent the sections best. The 'k' should be adjusted when setting up the algorithm so that the performance is above a certain threshold (Uğuz 2011). It is a simple algorithm, in fact there is not much learning except the use of training data.

Either all words in a document or a selection of words can be used as features for the kNN technique. Using the example documents index from the Naive Bayes method, as visualized again in Table 2.10, the class of the test document can be determined based on the 5 training documents. The most common method of determining the distance between the features of the training documents and the document to test is the Euclidian distance (Feldman & Sanger 2007), which is the square root of the sum of the differences squared for each individual term:

$$\sqrt{\sum_1^n (t_{n,w1} - t_{new,w1})^2})$$

Table 2.10: kNN example - Documents index

Terms:	*A340*	*787*	*A330*	*747*	*Flight*	*Wing*	*Manual*	**Class**
d_1	3		2		2	2		YES
d_2		4		5	7	2		NO
d_3	5		4		4		1	YES
d_4	3	1			2	4	4	YES
d_5		2			3		5	NO
d_{new}	2						2	?

Table 2.11: kNN example - Euclidian distance

Terms:	*A340*	*787*	*A330*	*747*	*Flight*	*Wing*	*Manual*	**Total**
d_1	$(3-2)^2$	$(0-0)^2$	$(2-0)^2$	$(0-0)^2$	$(2-0)^2$	$(2-0)^2$	$(0-2)^2$	$\sqrt{17} = 4.1$
d_2	$(0-2)^2$	$(4-0)^2$	$(0-0)^2$	$(5-0)^2$	$(7-0)^2$	$(2-0)^2$	$(0-2)^2$	$\sqrt{102} = 10.1$
d_3	$(5-2)^2$	$(0-0)^2$	$(4-0)^2$	$(0-0)^2$	$(4-0)^2$	$(0-0)^2$	$(1-2)^2$	$\sqrt{42} = 6.5$
d_4	$(3-2)^2$	$(1-0)^2$	$(0-0)^2$	$(0-0)^2$	$(2-0)^2$	$(4-0)^2$	$(4-2)^2$	$\sqrt{26} = 5.1$
d_5	$(0-2)^2$	$(2-0)^2$	$(0-0)^2$	$(0-0)^2$	$(3-0)^2$	$(0-0)^2$	$(5-2)^2$	$\sqrt{26} = 5.1$

So, if all words would be identical between two documents, the Euclidian distance is zero as all differences are zero. In this case, the differences are between 4.1 (d_1) and 10.1 (d_2). The last step is determining the class of the test document d_{new}. For this example, k=1 and k=3 can be used (k=5 cannot be used as there are only 5 training documents in total):

- k=1 results in YES as only the closest document is chosen which is d_1 with class YES
- k=3 results in YES as the closest ones are YES, YES and NO and the highest count 'wins'

In case this technique is used, the 'k' should be chosen beforehand, but in this example the outcome would have been YES either case (as was found with the Naive Bayes method). Additionally, this example can be scaled up to an entire document or section and multiple training and test documents.

Vector space: Support Vector Machines (SVM)

Another technique that uses the vector space to determine categories is the Support Vector Machines (SVM) method. The SVM method tries to find a decision surface between two classifications as is visualized in Figure 2.19 (Sebastiani 2002). The learning algorithm takes the outmost dots for the red and blue class and draws a line between them so that a new point (section) can be classified based its location with respect to the line. This is a simple example with only two classes and a two-dimensional space. Although this linear case is often used, it can be extended to higher dimensions and more classes as well. There are more options for determining the line between the classes, such as polynomials (Collobert et al. 2011). Additionally, data transformations can be applied so that a linear case is created or multiple support vectors (lines) can be drawn.

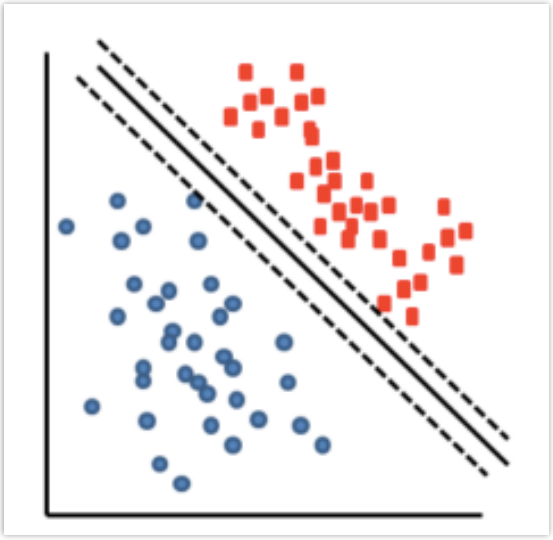


Figure 2.19: SVM example (Small & Medsker 2014)

Both the kNN and SVM method can use feature vectors that are transformed from the original (key) words in sections (Collobert et al. 2011), as was shown in the kNN example with the Euclidian distance feature vector. Additionally, a two-dimensional vector can be created from the original words by selecting specific words from a text or applying a kernel function to create a dot in a two dimensional space, basically a transformation from a multiple words feature vector to a two-dimensional vector (Nadkarni et al. 2011). Besides the Euclidian distance, common feature extraction methods are document frequency, information gain, mutual information, χ^2 squared and term strength (Yang & Pedersen 1997). These methods are difficult and time-consuming to implement and therefore usually only applied to improve the basic functionality of a technique, i.e. as an iteration to improve the performance of the technique.

2.8. Planned processing techniques: Text Clustering

The last section about techniques is related to text clustering techniques. Text clustering is defined as “the process of grouping text based on common attributes, e.g. all text related to corporate takeovers would form in a single cluster” (Small & Medsker 2014). The functionality of text clustering is comparable to classification as can be seen in Figure 2.20. The major differences lie in the goal of the process, whereas text classification aims to learn from training data to assign predefined features to the sections, text clustering does not need training and attempts to learn clusters for the sections by itself. It is therefore an unsupervised learning task in the machine learning domain (Liao et al. 2012). Text clustering can be used to assign previously unknown features to sections that can help in the ad-hoc processing or information retrieval, i.e. by determining the match of documents on other features than known words. For example, if a section is about the Airbus A330 but the section does not mention those words, the clustering learning algorithm can learn from other sections based on other specific words that it still belongs to the A330 group/cluster. However, text clustering techniques cannot be trained with training data, making the outcome of the process unpredictable beforehand, which is the exact definition of a clustering task.

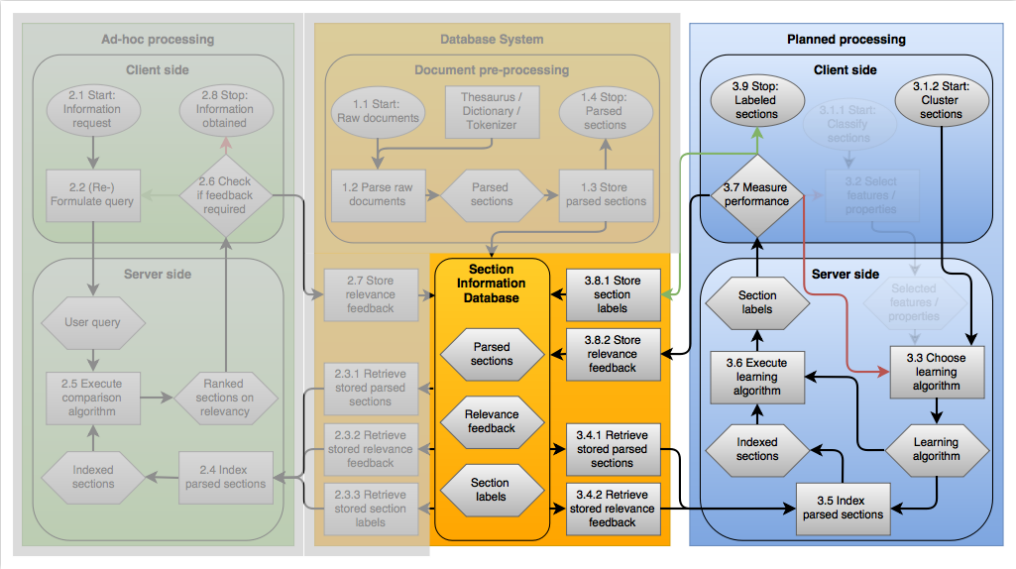


Figure 2.20: Clustering functions highlighted in the FFD

Unlike text classification, there are no specific tasks related to text clustering, therefore the rest of this chapter only focusses on a discussion of major text clustering techniques available in literature. There are two groups that will be discussed: hierarchical methods and partitional methods (Jain et al. 2000). The main differences between those methods is that hierarchical methods create a hierarchical structure of the data and clusters in levels in the hierarchy. On the other hand, partitional methods do not use a hierarchy and simply try to find clusters on only 1 level.

Hierarchical clustering methods

The hierarchical clustering methods construct a hierarchical structure of the data points available. In Figure 2.22, an example of 8 points are clustered used a hierarchical method. The clustering algorithm produces a hierarchical clustering structure of the points as can be seen in the dendogram in Figure 2.21. There are two major hierarchical clustering algorithms: single-link and complete-link algorithms (Jain et al. 2000). The single-link algorithms look at the minimum distance between points to measure similarity on which is bases the clusters, while the complete-link algorithms use the maximum distance between points to measure similarity. In practice, this means that single-link and complete-link systems can detect other structures in the data. The difference between these systems is clearly illustrated in Figure 2.23. The single-link and complete-link algorithm recognize both the clusters d1-d2, d3-d4, d5-d6, d7-d8 but on a higher level in the hierarchy a difference exists. The single-link algorithm on the left calculates the distance between d2-d3 and d2-d6 (minimum distance with points from other clusters) and recognizes that d2-d3 is the smallest thus have higher similarity. The complete-link algorithm on the right calculates the distance between d1-d4, d1-d6 and d1-d8 (maximum distance with points from different clusters) and recognizes that d1-d6 is the smallest thus have higher similarity.

Both methods have their advantages and it should be tested in practice with data at hand to see which method performs the ‘best’ in a specific scenario. This is however a tricky statement, as clustering in general is a method that is made to recognize previously unknown structures in data. Therefore, it cannot be easily said what is ‘best’, but results from clustering can be measured just like other techniques explained in previous chapters.

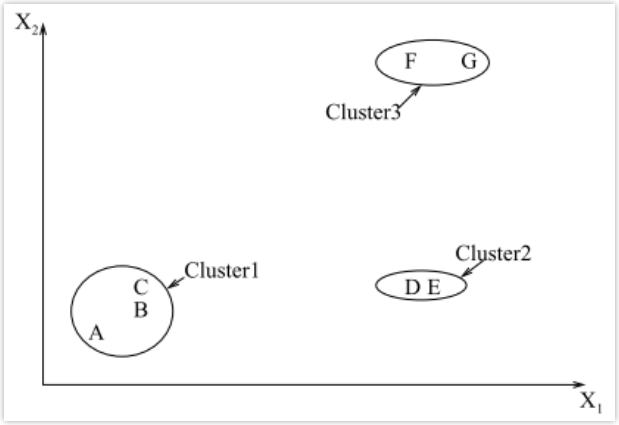


Figure 2.22: Points in three clusters (Jain et al. 2000)

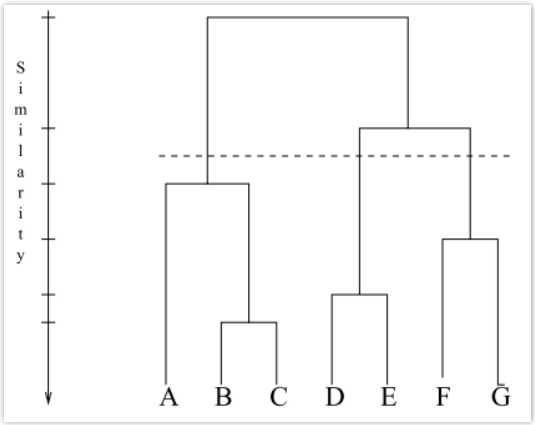


Figure 2.21: Dendogram for the clustering example (Jain et al. 2000)

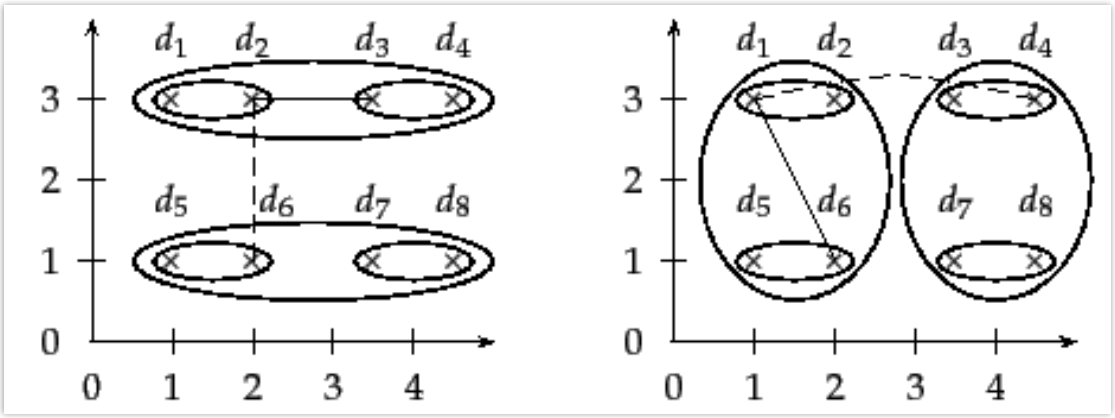


Figure 2.23: Single-link (left) and complete-link (right) clustering (Manning et al. 2009)

Partitional clustering

The second batch of clustering methods does not use a hierarchical structure to determine clusters, but uses a flat structure and is therefore also called flat clustering (Manning et al. 2009). There are two types of partitional clustering methods: hard clustering methods such as the K-means method and the soft clustering methods such as fuzzy c-means methods. The difference between those is that hard clustering assigns 1 cluster to a data point where as soft clustering assigns a fuzzy value (between 0 and 1 based on similarity) to a data point so that a data point can be assigned to multiple clusters (Jain 2010). The fuzzy c-means method is an extension to the K-means method, just as the fuzzy information retrieval technique is to the Boolean measure (from 0 or 1 to between 0 and 1).

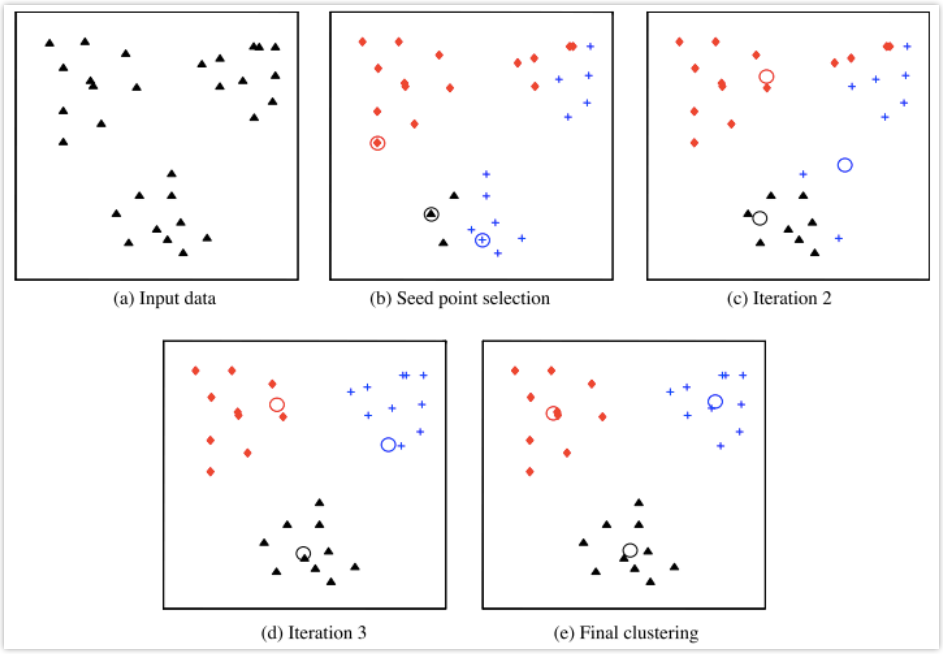


Figure 2.24: Example of K-means algorithm (Jain 2010)

The basic functionality of the K-means method is demonstrated in Figure 2.24. The task is to cluster the input data as can be seen in (a). First the algorithm randomly chooses seed points based on the number of clusters specified by the user (b) and from there onwards it iterates (c) and (d) till it converges to its final state (e). The iterations are performed by determining the distance between the points around the seed point and including and excluding new points based on proximity, and afterwards moving the seed point to the new center of the cluster (Jain 2010). In other words, the functionality of this hard clustering method is based on dividing the data space in three sections, one for each cluster.

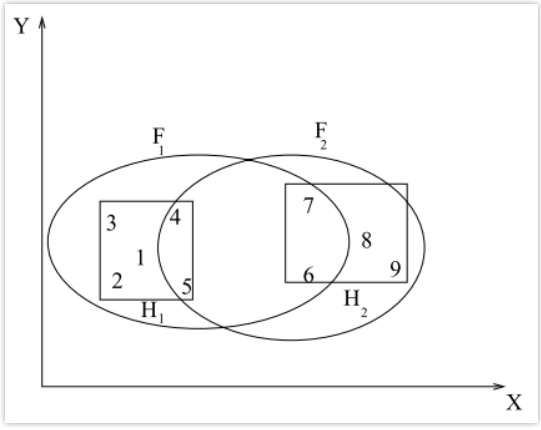


Figure 2.25: Fuzzy clustering example (Jain et al. 2000)

The fuzzy c-means gives another layer of information on top of the K-means method, a fuzzy weight to each data point indicating their likelihood to match the rest of the cluster. As the fuzzy c-means method is an extension of the K-means method, the functionality is the same but the outcome is a fuzzy weight assigned to each point based on their likelihood that they match the cluster, as well as multiple clusters assigned to the same point (Jain et al. 2000). This can be seen in Figure 2.25, where for example point 4 will get a high weight for cluster 1 and a low weight for cluster 2.

The performance of all clustering methods stems from the selection of right features in the data (sections) or the right transformations of the data, so that they fit in a X-dimensional space. A comparison of the two main sets of methods shows that partitional methods are more efficient than the hierarchical methods, but the hierarchical methods store more information (Jain et al. 2000).

2.9. Synthesis of techniques

This last sections have provided an overview of techniques to identify and extract information from text in general. Three distinct groups with techniques are identified in a functional flow diagram: document pre-processing (converting the raw documents to an accessible format), ad-hoc processing (query search through the sections / documents) and planned processing (classification and clustering). Each group contains a variety of techniques, as synthesized in Figure 2.26. It should be noted that in the document pre-processing part no techniques but tasks are described, as the techniques required to perform those tasks are often named similar and have a very similar process. For example, the task tokenization can be performed with the techniques sentence or word tokenization.

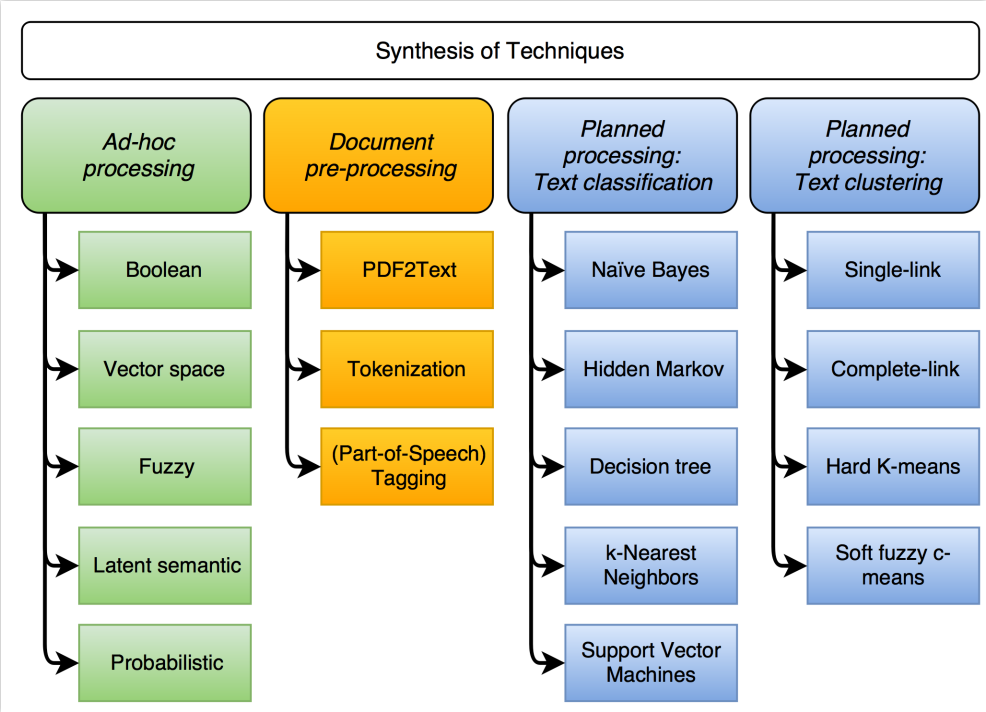


Figure 2.26: Synthesis of the techniques described in this literature review

2.10. Application of techniques in specific domains

In the previous sections, techniques are described for fulfilling the functions of document pre-processing, ad-hoc processing and planned processing (classification and clustering). In this section, an overview is presented of the application of these techniques in a domain-specific context. Three leading domains are identified in literature that use a mixture of these techniques for various tasks: medical science, biology and manufacturing. In each of the subsections an overview is given of the state-of-the-art of the domain, the methods used and their performance. A separate literature review could be made on each of these domains, so to limit the information presented only the top-level state-of-the-art and the most important methods are discussed.

2.10.1 Medical science

The first identified leading domain is related to medical science, which in literature is also called Medical Informatics (Afantenos et al. 2005). The medical science domain is characterized by very early integration of methods and techniques for identification and extraction of useful information from medical documents, either for the summarization of documentation or the retrieval of specific documents or sentences from documents to aid a doctor or aid in research. The reason for the use of automated systems is an information overload and rapid increase of clinical data over the years (Batet et al. 2011). Many documents are also now digitally available which means these technologies can be used.

Methods

The medical domain is characterized with tasks of named entity recognition (NER) and finding related information to the named entities (Doan, Bastarache, Klimkowski, Denny, & Xu, 2010), e.g. drug names in clinical notes or recognizing gene names. Most of the researches use MetaMap as a tool, which has a dictionary with medical names and their related names to be able to recognize them in text (Spasic et al. 2014). Typical precision is around 55%, recall is between 55-59% and F-scores are between 64-66% when the MetaMap is used for NER (Spasic et al. 2014). In addition, Jagannathan et al. (2008) evaluates 4 commercial NLP tools (Coderyte, Language and Computing, Artificial Medical Intelligence, LingoLogics) based on their ability to extract medical names and found precision scores between 98-100%, recall between 52-98% and F-score between 68-98%. If related information to the names was searched simultaneously, the scores changed to a precision between 71-97%, recall between 79-98% and F-scores of 80-90%. So, those commercial systems performed better than the open source MetaMap and form a baseline for NER standards in the medical domain. Another specific mention of a method to perform NER is using SVM (text classification technique), for which F-scores between 84-93% depending on the task was achieved (Jiang et al. 2011). It is a wide range of scores, which can be attributed to the difference in techniques and for example dictionaries used.

For most of the systems mentioned in literature, the specific techniques used are actually not mentioned or mentioned as a group of techniques applied in one method (Spasic et al. 2014). What can be learned from literature is that a domain-specific ontology, dictionary or taxonomy is important for coreference resolution and to aid the classification and retrieval tasks. Additionally, syntactic-based techniques perform worse than semantic-based techniques, as information is very domain-specific such as the use of abbreviations, so that semantic additional information adds to the understanding of the context of texts (Jiang et al. 2011).

2.10.2 Biology

The second leading domain is the biology or biotechnology. In this domain, most researched tasks are identifying and extracting information on biological entities, interaction of entities and the localization of entities (Natarajan et al. 2005). As in the medical domain, there was an explosion of information and data that lead to the use of automated techniques for identification and extraction of relevant information from documents (Hakenberg et al. 2004).

In contract to the medical domain, the biology domain mentions many more techniques and more generic tasks. For example, KDD is mentioned in the biology context and then specifically the data mining element of KDD (Brusic & Zeleznikow 1999). Furthermore, information retrieval gets attention specifically in the context of finding biological documents or passages that are relevant to a user (Hu et al. 2012).

Methods

The methods found in the biology domain literature are also more diverse than in the medical domain. However, NER is here also an important task. Natarajan et al. (2005) mention many studies using NER that reported precision scores between 70-100%, recall between 65-95% and F-scores are in the same range: between 70-90%. Most of those systems use a large corpus with biomedical information that is used for coreference resolution and NER, such as GENIA, BioCreative Task1A or Yapex. The use of a proper dictionary is very important, as the scores vary greatly per dictionary for the same task (Hatzivassiloglou et al. 2001).

Additionally, as a next step after NER relevant sections to a user information need can be identified and extracted (information retrieval). Typical scores of recall and precision are between 60-95% (Hao et al. 2005), again dependent on which the corpus and dictionary used. On the planned processing side, also text classification methods have been applied in this domain using techniques such as Naive Bayes or vector space techniques. For those methods the precision is relatively high compared to recall, with scores of 60-80% and 20-30% respectively (Craven & Kumlien 1999; Stapley et al. 2001). In rare cases, systems reach over 90% precision and recall, which is possible by more relevant training data than usual and a very relevant corpus or taxonomy for the task performed (Hakenberg et al. 2004).

2.10.3 Manufacturing

The last leading domain is the manufacturing domain, which is a broader domain that covers both the use of learning techniques, e.g. for fault diagnosis, but also the more relevant use of these techniques for finding information in documents (Rajpathak 2013). The manufacturing domain is more characterized by a descriptive view on the use of these techniques and methods than the actual use of those methods (Choudhary et al. 2009) in a similar fashion as the aircraft maintenance domain showing a more top-level than practical view. On the other hand, the manufacturing started using databases and statistical techniques as early as the 1980s (Harding et al. 2006). Only in the context of discovering knowledge from document this only started more recently, as with the medical and biology domains.

Methods

The methods described in the manufacturing papers provide information on the techniques used, but often do not give the precision and recall scores (Choudhary et al. 2009). It is described that classification techniques are applied such as Bayesian and support vector methods, as well as clustering methods such as K-means or fuzzy-c means. For example, Caramia & Felici (2006) mention a method to identify information from pages with semantic techniques based on decision trees, but they do not provide any precision, recall or F-scores. Additionally, Crespo & Weber (2005) describe the use of a fuzzy c-means method for clustering related to customer segmentation, but also do not provide any performance measures such as precision or recall. Lastly, Rajpathak (2013) describes an ontology-based text mining system (clustering) that uses the ontology to semantically extract relevant information from texts and clusters texts accordingly. Texts are clustered based on frequency of co-occurring terms and the following performance measures are obtained: between 83-99% for precision, recall between 90-99% and F-score between 90-97%. These different performances are for different terms that were searched in the text and used for clustering. Differences in the performance measures come from the use of different taxonomies or dictionaries for the tasks (Choudhary et al. 2009).

2.11. Conclusion: Closing the Research Gaps

The last sections have provided an overview of applications to use digital documentation in the domains of aircraft maintenance, medical science, biology and manufacturing, as well as techniques to identify and extract information from text in general. While for most domains an abundance of literature is available on these topics, the aircraft maintenance domain lacks in-depth literature about the identification and extraction of information from digital maintenance documentation. Two research gaps were identified: the lack of literature related to automated systems with this purpose as well as techniques that could be used in such a system. Despite these research gaps, the literature that is available indicates a need for a digital automated maintenance documentation system.

By reviewing the generic domains for techniques and the specific domains for the application of these techniques, the research gaps can be closed from a theoretical perspective. First, this literature review combines research performed in six generic and three specific domains related to the identification and extraction of information from text and places it in the context of the aircraft maintenance domain. It extends the literature in the aircraft maintenance domain by providing an overview of techniques available in these nine domains to create a system for the automated contextualization of maintenance documentation.

Moreover, this review does not only provide an overview of the research performed in these nine domains, it goes one step further and combines the functionalities available in each specific domain to create an overarching functional flow diagram for the entire process of identification and extraction of relevant information from documents. The KDT domain describes the combination of 3 research domains in the process of knowledge discovery in text, but in this domain, there is no literature available that describes the processes completely for identification and extraction of information from documents or text. In that regard, this literature study also extends the KDT domain by providing a complete overview of processes available in all these generic domains to perform the described task. Combining these functionalities in a comprehensive overview from all domains is a novel approach in literature. Additionally, the synthesis of techniques demonstrates how some of these functions could be executed from a theoretical perspective.

Knowing a theoretical system and theoretical techniques / approaches is one step towards answering the research question, but a comparison between these techniques could not be made based on the literature. An important finding of the domain-specific literature is that a reliable comparison of techniques cannot be made as the performance is highly dependent on the task, taxonomy/dictionary, documents and, for example, the amount and quality of training data available for text classification. Nevertheless, a rough indication of the performance of techniques can be extracted from literature. For simple techniques, such as word detection, the performance (precision, recall and F-score) can get very close to 100%. When techniques become more complex, such as co-reference resolution, the performance drops to around 90%. More advanced techniques based on semantics even have a lower performance, down to 80% on average. These percentages are an indication of the mean performance of different techniques and, as mentioned before, there is high deviation from this mean depending on the research setting.

Additionally, due to the varying performance of different techniques, it is found that for an efficient and reliable automated system a combination of techniques is necessary. Document pre-processing techniques are required to transform the raw documents to sections and ad-hoc processing are necessary to select the right sections to match the information request of the user. Supporting functions of such a system are text classification and text clustering, as these planned processing methods can speed up the ad-hoc processing by adding metadata to documents. Reflecting back on the research objective of this thesis, which is to find a recommended approach to use in an automated system, a theoretical answer is to combine techniques in specific elements of the system as depicted in the developed FFD and synthesis of techniques.

The next step is to create a prototype automated system based on the FFD and a test environment to compare the different techniques depicted in this chapter. This is addressed in the next chapter of this thesis, the methodology for testing the techniques.

3. Methodology

3.1. Research Design

Chapter 2 provided a comprehensive theoretical FFD that can be used to design an automated documentation system and a set of techniques that can be applied to the functions in such a system. The research objective is to formulate recommendations for how to extract and identify information from maintenance documentation. By creating a prototype automated system and testing different techniques in this system, a recommended approach can be formulated to accomplish the objective. As no systems are available in literature, a new system is created from scratch. This novel system encompasses document pre-processing, two databases, a live search (ad-hoc processing) and testing of techniques (planned processing), as shown in Figure 3.1. The strength of the system are the connections between the different elements. The document data is stored in two databases and can be accessed by the live search and altered by the testing functions of the system. In other words, after performing tests, the results can be stored in the databases and immediately be used in the live search of the system. Each of the functions in the system is developed so that they can be extended and customized.

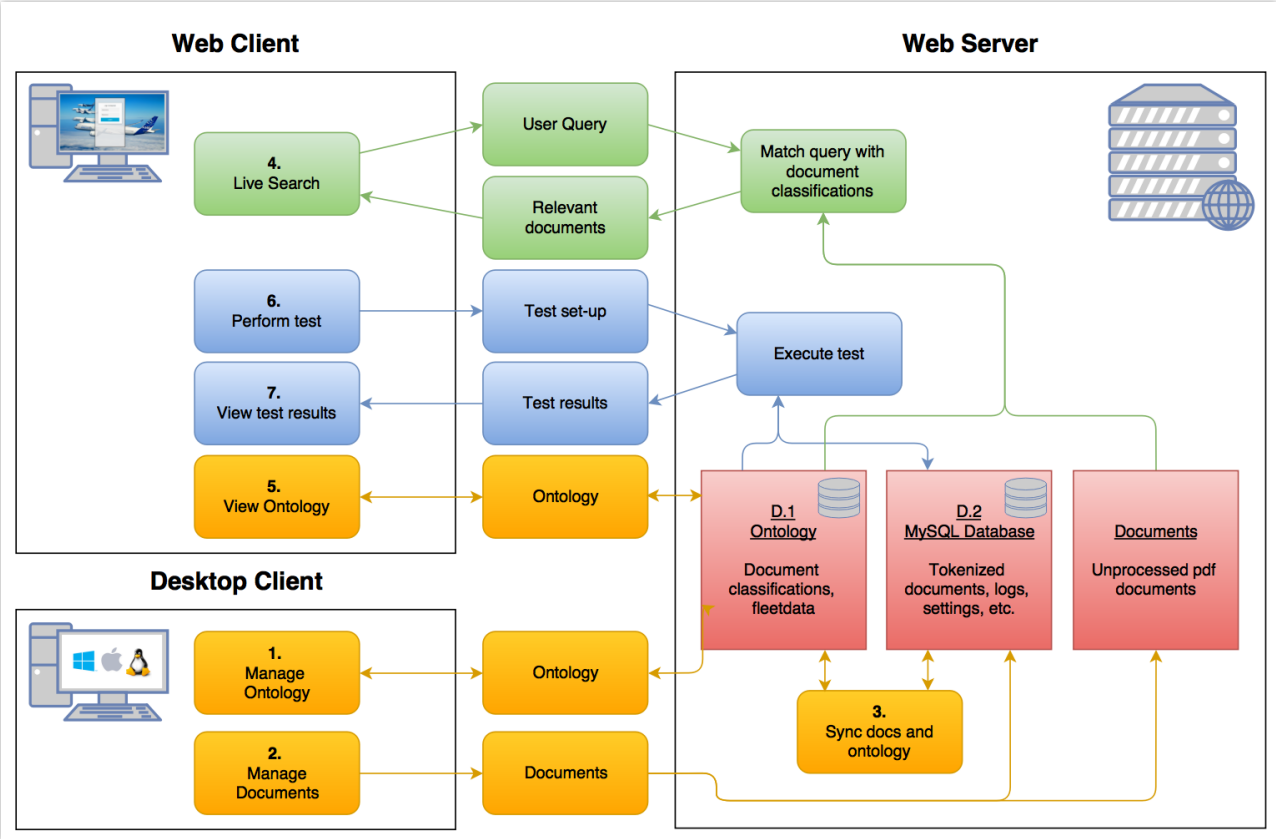


Figure 3.1: Top-level system overview

The developed generic system is then altered to work with specific maintenance documents so tests can be performed. As is found in Chapter 2, the performance of techniques is highly dependent on the data used. Therefore, the first step in this research design is to determine the data that is used in the system. A fully operational automated system would use multiple types of maintenance documentation, e.g. using Airworthiness Directivess (ADs), Aircraft Maintenance Manuals (AMMs) or Service Bulletins (SBs). However, this is just a prototype system designed to determine a

recommended approach for such a large system. As techniques must be designed specifically for a task, a selection is made for only one data source rather than designing techniques for multiple data sources, so that more tests on one document type can be performed. The only requirement for the data source is that a large amount of information can be obtained to obtain significant test results. There is one type of document that is available in the public domain, so it can be freely downloaded, and it is available in a large quantity: Airworthiness Directivess (ADs). A total of 14008 AD documents can be freely downloaded on the Internet from an EASA tool. Section 3.2 discusses in detail the data collection, cleaning and selection.

After establishing the data source for the system, the other functions of the generic system can be designed for this data source. The live search and testing of techniques are based on a use case: given an aircraft registration mark and a topic based on an ATA chapter, identify the ADs that are applicable / relevant. To implement this use case, an aircraft fleet is used to have tangible data to search for in the documents. The specific aircraft fleet does not influence the functionality of the system, so an arbitrary fleet was chosen: the TAP fleet. From the TAP fleet a selection of aircraft is made and based on their aircraft registration marks the live search can be performed. The document classifications and fleet data used to provide an answer to the query is stored in the ontology. Additionally, the unprocessed AD pdfs are stored on the server so that these can be send back to a user based on the query. The results for the live search are presented based on Boolean relevancy: either an AD is relevant for the specific aircraft and ATA chapter or it does not. This is further elaborated in section 3.3, which is an overview of the system and its functions.

The relevancy of the ADs to these two inputs is determined by using the testing of techniques, which is the core of this research. The tasks that are tested are based on the information need to match the registration mark and ATA chapter from the live search to the ADs. Seven document properties that are relevant to matching the documents were included in the system: Publisher, Language, Type (EAD/AD), Superseded, Aircraft Manufacturer (stated in the document), Aircraft Type and ATA chapter. The first five of these properties are used to test different techniques on as these have a large sample size in the dataset, the Aircraft Type and ATA chapter can only be executed with the Boolean as the sample size is too small for text classification methods based on Machine Learning, so these are not tested. The detailed set-up for the tests, i.e. the properties and techniques used, is provided in section 3.4 of this chapter. A test-setup is a combination of a property with a set of techniques, for example determining the language used in a document (English or French).

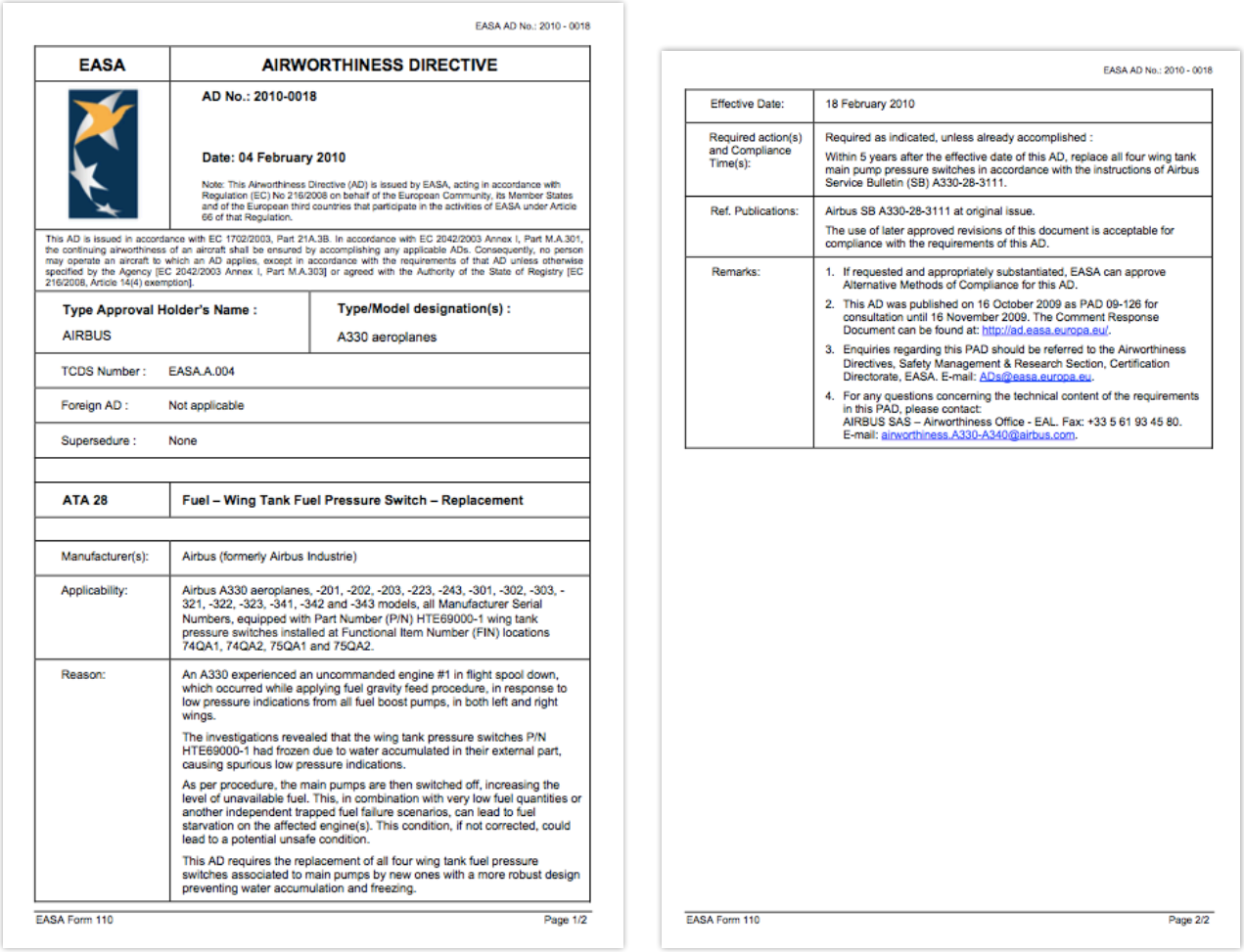
An important element of testing and comparing the techniques is knowing the performance of each technique. To determine this, the actual value for each property is determined manually combined with EASA metadata for the ADs. For example, the publisher of an AD can be determined based on metadata provided by the EASA tool. The verification of the test scripts and the validation of this data is discussed in section 3.5. The verification demonstrates the correct application of the tests in the test environment by re-creating the test set-up in another software program. Additionally, the validation of the results is to determine whether the actual values for all properties are correct, which is performed by using manual checks and inverse application of the techniques to find outliers.

Having determined a correct execution of the system, the final element of the methodology is a pre-test sensitivity analysis as discussed in section 3.6. There are some important variables besides the properties and techniques that have an impact on the results of the tests, such as the amount of training data used in the tests and the sample size. In that section, each of these variables is tested by keeping all other variables the same and varying only this variable. At the end, a selection is made for the final test set-ups and parameters for the tests.

3.2. Data Collection, Cleaning and Selection

As the system and the performance of the techniques are highly dependent on the data used, this section first describes the data that is used in the system and the tests. ADs are chosen as the data source for this system due to their availability in the public domain so they can be freely downloaded. Additionally, the only requirement for the data source is that a large dataset is used to obtain significant test results, which is the case for ADs.

An AD is a notion from the relevant regulatory body to an aircraft manufacturer that there is a safety deficiency with one of its aircraft which affects the airworthiness of this aircraft, sometimes accompanied with instructions with how to fix the issue[1]. A typical AD can be viewed in Figure 3.2, an Airworthiness Directivess from the publisher EASA. Important information from the AD is the publisher, language, issue date, effective date, manufacturer, applicability, ATA chapter and reason of publishing the document. These ADs can be freely downloaded in the Safety Publications Tool of EASA[2].



EASA AD No.: 2010 - 0018

EASA	AIRWORTHINESS DIRECTIVE
	AD No.: 2010-0018 **Date: 04 February 2010** Note: This Airworthiness Directive (AD) is issued by EASA, acting in accordance with Regulation (EC) No 216/2008 on behalf of the European Community, its Member States and of the European third countries that participate in the activities of EASA under Article 66 of that Regulation.

This AD is issued in accordance with EC 1702/2003, Part 21.A.3B. In accordance with EC 2042/2003 Annex I, Part M.A.301, the continuing airworthiness of an aircraft shall be ensured by accomplishing any applicable ADs. Consequently, no person may operate an aircraft to which an AD applies, except in accordance with the requirements of that AD unless otherwise specified by the Agency [EC 2042/2003 Annex I, Part M.A.303] or agreed with the Authority of the State of Registry [EC 216/2008, Article 14(4) exemption].

Type Approval Holder's Name :	Type/Model designation(s) :
AIRBUS	A330 aeroplanes

TCDS Number :	EASA.A.004
Foreign AD :	Not applicable
Supersedure :	None
ATA 28	**Fuel – Wing Tank Fuel Pressure Switch – Replacement**
Manufacturer(s):	Airbus (formerly Airbus Industrie)
Applicability:	Airbus A330 aeroplanes, -201, -202, -203, -223, -243, -301, -302, -303, -321, -322, -323, -341, -342 and -343 models, all Manufacturer Serial Numbers, equipped with Part Number (P/N) HTE69000-1 wing tank pressure switches installed at Functional Item Number (FIN) locations 74QA1, 74QA2, 75QA1 and 75QA2.
Reason:	An A330 experienced an uncommanded engine #1 in flight spool down, which occurred while applying fuel gravity feed procedure, in response to low pressure indications from all fuel boost pumps, in both left and right wings. The investigations revealed that the wing tank pressure switches P/N HTE69000-1 had frozen due to water accumulated in their external part, causing spurious low pressure indications. As per procedure, the main pumps are then switched off, increasing the level of unavailable fuel. This, in combination with very low fuel quantities or another independent trapped fuel failure scenarios, can lead to fuel starvation on the affected engine(s). This condition, if not corrected, could lead to a potential unsafe condition. This AD requires the replacement of all four wing tank fuel pressure switches associated to main pumps by new ones with a more robust design preventing water accumulation and freezing.

EASA Form 110 Page 1/2

EASA AD No.: 2010 - 0018

Effective Date:	18 February 2010
Required action(s) and Compliance Time(s):	Required as indicated, unless already accomplished : Within 5 years after the effective date of this AD, replace all four wing tank main pump pressure switches in accordance with the instructions of Airbus Service Bulletin (SB) A330-28-3111.
Ref. Publications:	Airbus SB A330-28-3111 at original issue. The use of later approved revisions of this document is acceptable for compliance with the requirements of this AD.
Remarks:	1. If requested and appropriately substantiated, EASA can approve Alternative Methods of Compliance for this AD. 2. This AD was published on 16 October 2009 as PAD 09-126 for consultation until 16 November 2009. The Comment Response Document can be found at: http://ad.easa.europa.eu/. 3. Enquiries regarding this PAD should be referred to the Airworthiness Directives, Safety Management & Research Section, Certification Directorate, EASA. E-mail: ADs@easa.europa.eu. 4. For any questions concerning the technical content of the requirements in this PAD, please contact: AIRBUS SAS – Airworthiness Office - EAL. Fax: +33 5 61 93 45 80. E-mail: airworthiness.A330-A340@airbus.com.

EASA Form 110 Page 2/2

Figure 3.2: A typical Airworthiness Directive (2010-0018); page 1 and 2

1 EASA, https://www.easa.europa.eu/easa-and-you/aircraft-products/airworthiness-directives-ad (18-11-2016)

2 EASA Safety Publications Tool, http://ad.easa.europa.eu (12-08-2016)

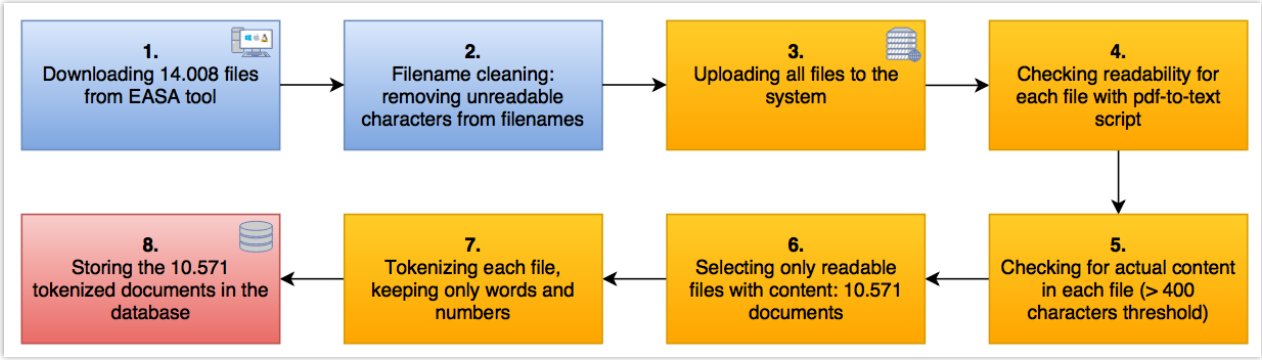


Figure 3.3: Data Collection, Cleaning and Selection process

The generic system as described in section 3.1 is adjusted to work with the ADs. The entire process from downloading the ADs from the EASA tool up until storing them in a MySQL database is visualized in Figure 3.3. The system and database are described in much more detail in the next section and are therefore not elaborated here. The first step of the process is downloading the files from the EASA tool. All ADs up to 12-08-2016 (the oldest is from 1957), which leads to a total of 12153 publications. Some publications have multiple documents, for example due to multiple translations, which leads to a total of 14008 pdf documents. The documents are from many different publishers as not only EASA documents are available, but also e.g. FAA and TransportsCanada ADs. Every publisher uses its own template for the AD, but the information is often very similar as there are regulations for what should be in an AD.

After obtaining the 14008 documents, the next step is cleaning the filenames, which involves checking them for unreadable characters. As the filenames are also used as a document identifier, the allowed characters are very strict. This resulted in the removal of characters in filenames such as spaces, parenthesis and certain double underscores. No files had to be removed from the dataset. After cleaning the filenames, the third step is uploading all files to the system. Subsequently, a script is executed that extracts the raw text from the pdf files based on the PDF2Text technique as described in the literature review. The documents range from 1957 up to 2016, which leads to that some documents cannot be read by the script because of older file formats and scanned documents instead of pdfs with the current standards. The results can be seen in Figure 3.4. There are 11710 documents that can be read and 2298 documents that cannot be opened are omitted for further processing. Next, the actual content of all documents is checked. Some documents have the text “If you require the AD, please e-mail X” and some are empty. These documents can be identified by looking at the amount of characters, which is found to be lower than 400. Including the documents that cannot be read by the PDF2Text script, there are 3437 documents that do not have content and 10571 that are readable and have content, as is visualized in Figure 3.5.

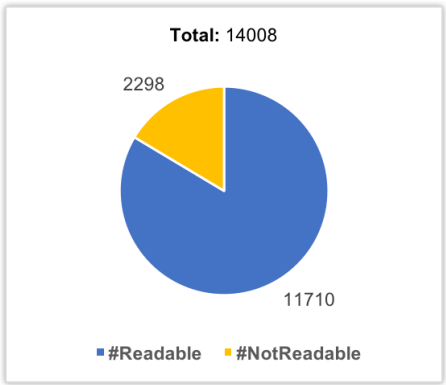


Figure 3.4: Document Analysis - Readability

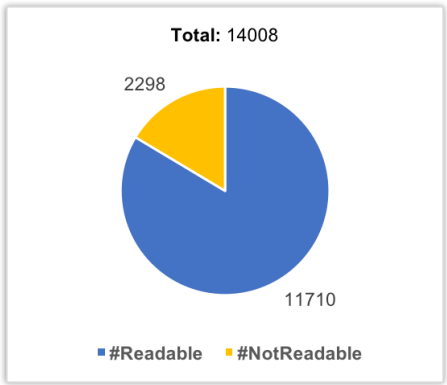


Figure 3.5: Document Analysis - Actual Content

At this point, the 10571 documents that are readable and have actual content are selected as the population for the system and subsequent testing. The two final steps are preparing the documents for database storage. A tokenizer is applied on the remaining documents, which extracts only words (tokens) from the documents. The tokenizer is very simple: split a text string (the document) on every whitespace. Before tokenization can be performed, first a Regular Expression script is executed on each document: /[^(a-zA-ZA-Яа-я0-9_)+\s]/g. The Regular Expression is used to remove all unwanted characters from a text string (the document). In this case, only letters, numbers, special characters such as ß and a few other characters such as _ and + are kept. This is performed to remove e.g. dots and commas from the documents as these are unwanted for tokenization. If this would not be performed, a token could become 'example.' and the words 'example' and 'example.' would not be counted together, but seen as two distinct tokens.

With the documents 'cleaned' and tokenized, the last step of the process is storing the 10571 tokenized ADs in a database so that they can be retrieved quickly for testing. The set-up of the database is discussed in the next section along with other components of the system.

3.3. Development of the System and Test Environment

With the data of the system defined, the system is tailored towards it. Figure 3.1 in the research design section provides a top-level overview of the system, with its main components the desktop, web client and web server. Before diving in detail in the functionality of the system, first the hardware/software used for the system components is described.

3.3.1. System Components and hardware / software

The generic set-up of a task is to identify the location of a piece of information in the ADs, extract this information and save it so that it can be quickly retrieved as an answer to a query from a live search. To establish possible hardware/software combinations in which this system can be created, first a set of requirements is made. The ADs are formatted as pdf documents so the hardware/software should be able to extract text from the pdf files. Furthermore, the software should be able to process the text from the pdfs and search through its contents. Lastly, a system with a server for processing and a client side that can be used by a user to search and retrieve information. Most of the processing software such as Matlab do not incorporate the set-up of a convenient client interface combined with separate processing, therefore another hardware/software combination is chosen: web client / server. By means of a web client, an interface can be set-up for user input and this can be combined with a server for processing. Web programming languages are also able to perform text processing.

Web Server

- VPS
- Ubuntu 14.04
- Node.js (JavaScript)
- Dedicated processing
- FTP and VNC connection with desktop
- WebSocket (live) connection with web client

The web server is a Virtual Private Server (VPS), as this does not interfere with other programs while running and it can be online 24/7. This means the server is on a server farm and dedicated to this test environment. The software on this VPS is Ubuntu 14.04 with very few programs installed. The most important for the test environment is the web server software. There are many options, but due to experience with node.js this web server software is chosen. A node.js server is based on

JavaScript, using Chrome's V8 JavaScript engine[3]. There are many advantages of this type of server, but for this research a relevant advantage is node.js' package ecosystem, npm, which is has the largest set of open source libraries in the world[4]. Libraries are available for e.g. extracting text from pdf documents (PDF2Text). The node.js server is combined with the WebSocket technology, which is a technology which allows for a dynamic secure two-way connection between the client and server[5]. This allows sending and receiving between client and server without having to continue sending requests to the server.

Web Client

- Accessible in any desktop and mobile browser
- Programmed using JavaScript, HTML and CSS
- Interface with the server for live search, ontology view and testing

As node.js is chosen, the primary scripting language is JavaScript, which is a lightweight and commonly used web programming language6. JavaScript is most well-known for the scripting of web pages, but it can also be used for non-browser environments such as node.js and can also be used for processing the techniques in this test environment. This means that except some HTML and CSS for the web client most of the code is programmed in JavaScript.

Desktop

- Any Operating System
- Web Client access via browser
- Protégé for ontology management
- Uploading docs and ontology via FTP
- VNC interface with the server

Additionally, VNC (remote viewing) and FTP (remote file access) software is installed on the VPS, which enables the upload/download of files and viewing the server remotely. The input data for the tests, the maintenance documents, can be uploaded by means of the FTP software.

Lastly, two databases are used to store all data in the system: a MySQL database and an ontology. The MySQL database is installed on the web server and is used for storing the tokenized documents as discussed in section 3.2 as well as the test results. The documents table in the database consists of the name and tokens in for each document, while the test results table has the test set-up and results for each test performed. A server script is created to connect the database with the other scripts, such as the live search and testing scripts. An example of the test results in the MySQL database can be seen in Figure 3.6.

MySQL database

- MySQL Workbench 6.0
- Two tables: documents and test results
- Interface created with the rest of the scripts

#	test_id	method	learning_setting	knn_number	property	documents	training	dimension_reduction	population_setting	docu
454	531	Boolean	No Continuous Learning	N/a	hasDocumentPublisher	Random 20%	None	None	Preselected Subsample	7437
455	532	Boolean	No Continuous Learning	N/a	hasDocumentLanguage	Random 1%	None	None	All docs	174
456	533	Boolean	No Continuous Learning	N/a	hasDocumentLanguage	100%	None	None	All docs	174
457	534	Boolean	No Continuous Learning	N/a	hasDocumentLanguage	100%	None	None	All docs	174

Figure 3.6: Part of the test results table in the MySQL database

3 Node.js Foundation, https://nodejs.org/en/ (18-11-2016)

4 NPM Inc., https://www.npmjs.com/ (18-11-2016)

5 Mozilla Developer Network, https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API (18-11-2016)

6 Mozilla Developer Network, https://developer.mozilla.org/nl/ (18-11-2016)

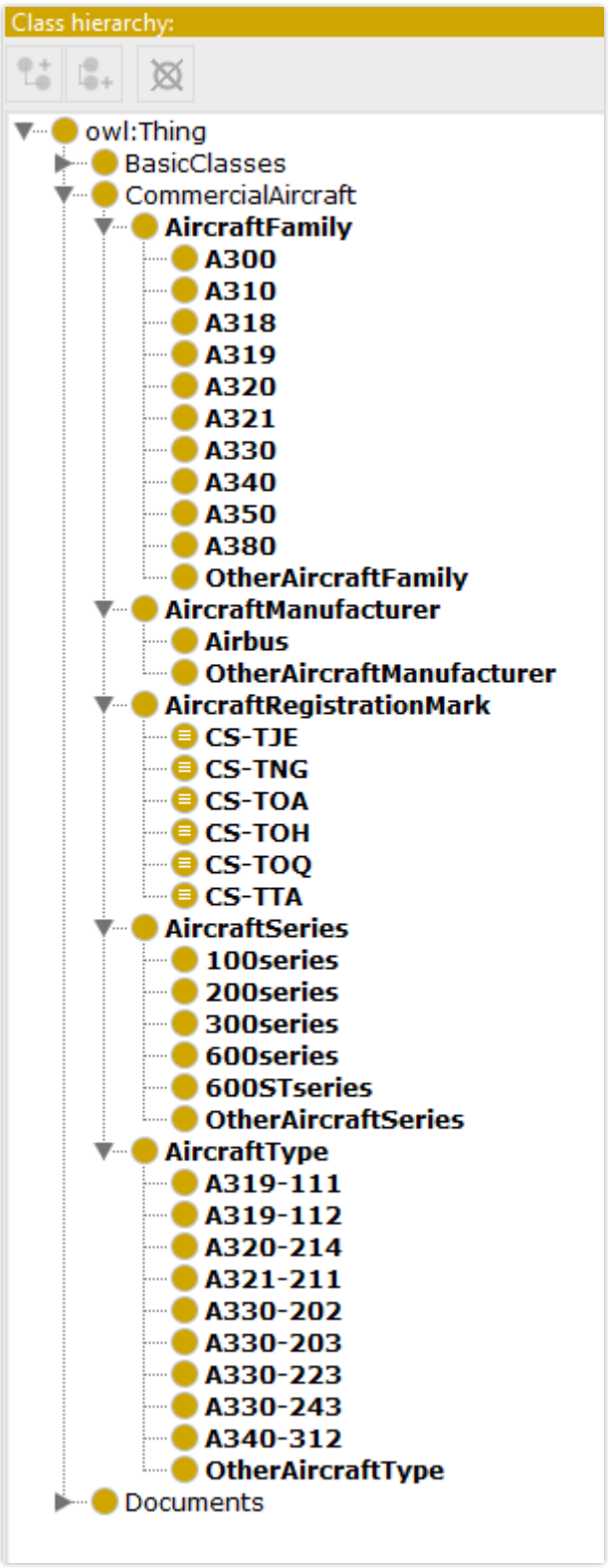


Figure 3.8: Fleet data in the ontology

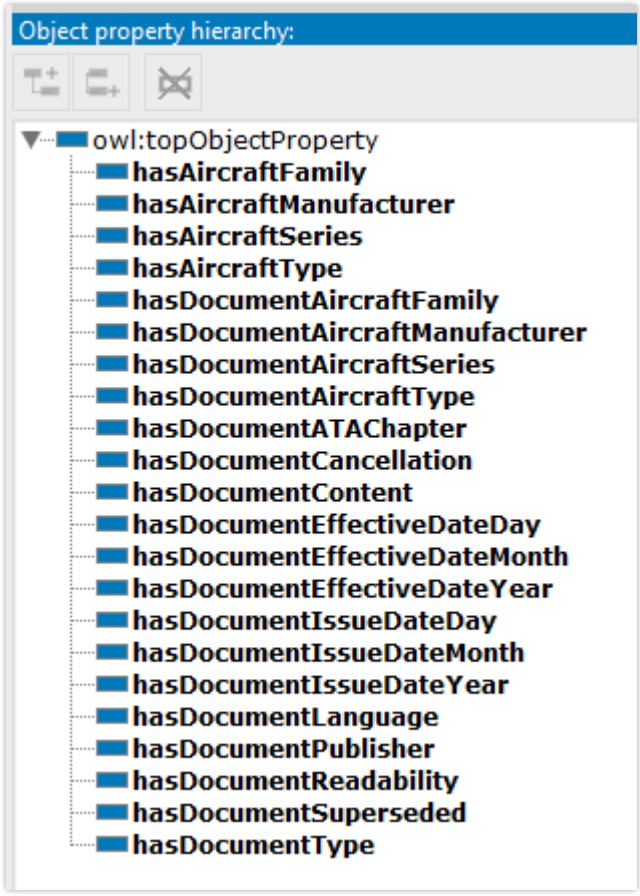


Figure 3.7: Properties in the ontology

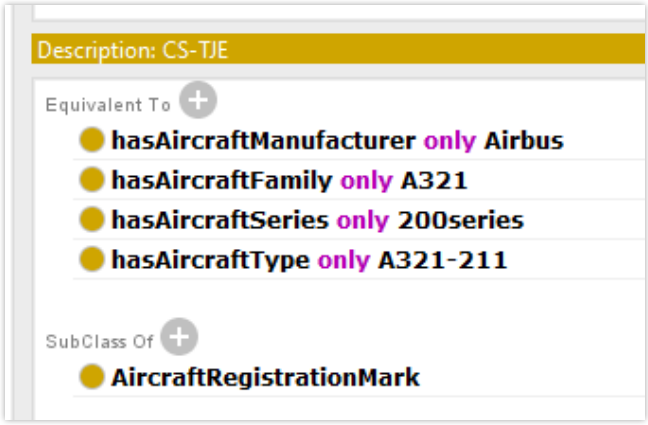


Figure 3.9: Information linked for registration mark CS-TJE

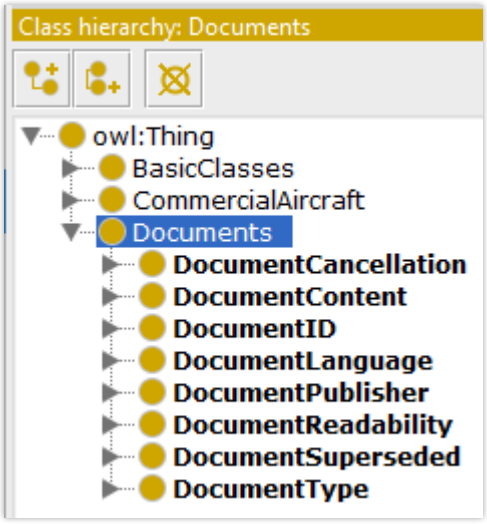


Figure 3.10: Document classes hierarchy in the ontology

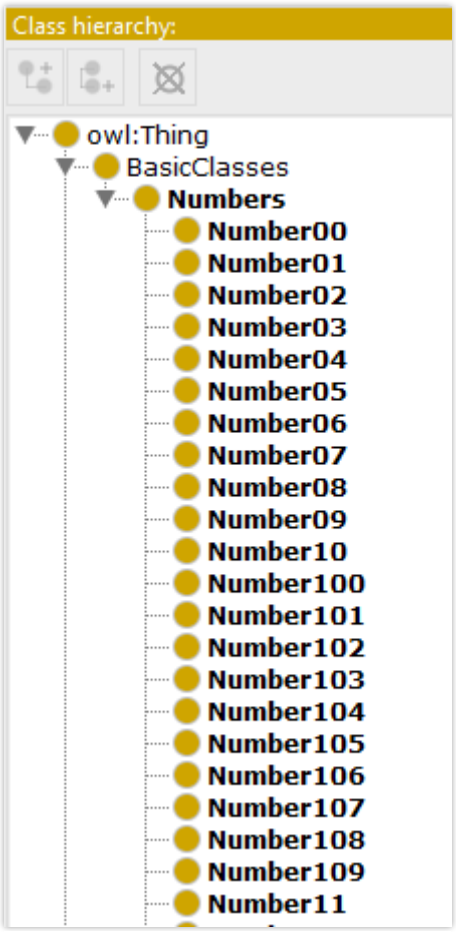


Figure 3.12: Basic Classes in the ontology

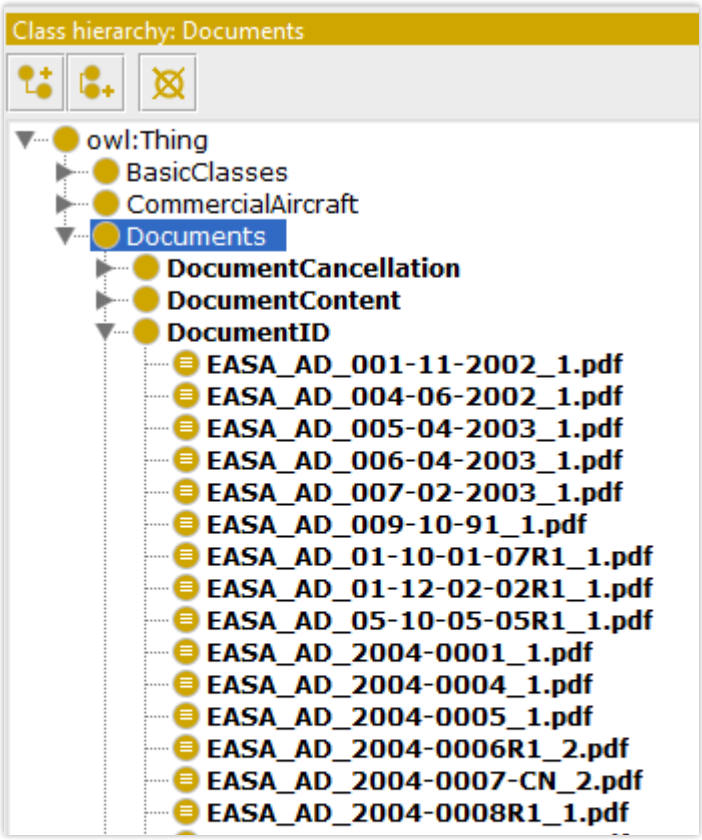


Figure 3.11: List of documents in the ontology

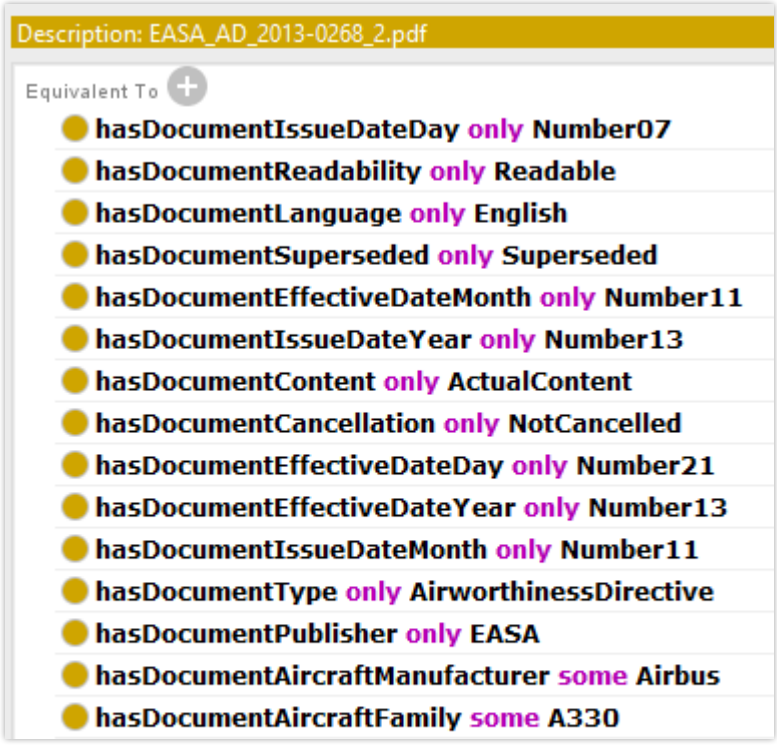


Figure 3.13: Document classifications for document EASA_AD_2013-0268_2.pdf

Ontology

- OWL / XML syntax
- Fleet data and doc. classifications
- Ontology interface script on the server
- Access via Protégé on the desktop

The other database is an ontology. An ontology is a more convenient method of storing linked data. Where the MySQL database is efficient in storing chunks of data in rows and columns, an ontology is used for describing relationships between data. It is stored in an OWL / XML data format, which is a similar format used for e.g. RSS feeds. The ontology uses (nested) classes and properties to describe the data. To use the data in the ontology on the server, a novel interface between node.js (or JavaScript in general) and OWL / XML files is created as no libraries were available in the npm database. This new module enables the reading, changing and writing / storing of an ontology file on the server. This means that when tests are performed and new document classifications are made, these can immediately be added to the ontology and viewed in the web client or on the desktop via Protégé. As an example of the scripting performed in this thesis, the ontology interface script can be viewed in Appendix 1.

The ontology is used for two datasets: the selected TAP fleet data and the document classifications which are obtained from the tests. There are six aircraft selected from the TAP fleet with the following registration marks: CS-TJE, CS-TNG, CS-TOA, CS-TOH and CS-TTA. These are inserted as classes in the ontology, as can be viewed in Figure 3.8. This information is supplemented with Aircraft family, manufacturer, series and type information. The information is linked together using properties, the total list of properties in the ontology can be viewed in Figure 3.7. This way of storing the information can be more easily extended than the MySQL database. Figure 3.9 shows the links of one registration mark, CS-TJE, with the other categories. In this way, the registration mark class CS-TJE is linked to other classes such as the Aircraft Type A321-211 using the property hasAircraftType.

The other dataset in the ontology, the document classifications, are organized in a similar fashion to the fleet data. As visualized in Figure 3.10, there are 8 groups of classes related to the documents: document ID (the unique document identifier as with the registration mark in the fleet data), document cancellation, document content, document language, document publisher, document readability, document superseded and document type. All these groups have their own property and additionally there are a few properties related to the Issue Date, Effective Date and ATA Chapter (hasDocumentATAChapter, etc.). These properties have a specific number, for example ATA 21. To store this information in an ontology, a branch of Basic Classes is created as can be seen in Figure 3.12, as all information should be in a class, which is different than a MySQL database. Therefore, the property hasDocumentATAChapter can be given the class Number21. The first part of the list of documents in the ontology can be viewed in Figure 3.11, as 14.008 documents would be too big a list to display. Finally, Figure 3.13 demonstrates how some of the properties are linked to the document EASA_AD_2013-0268_2.pdf, which is comparable to the fleet data case.

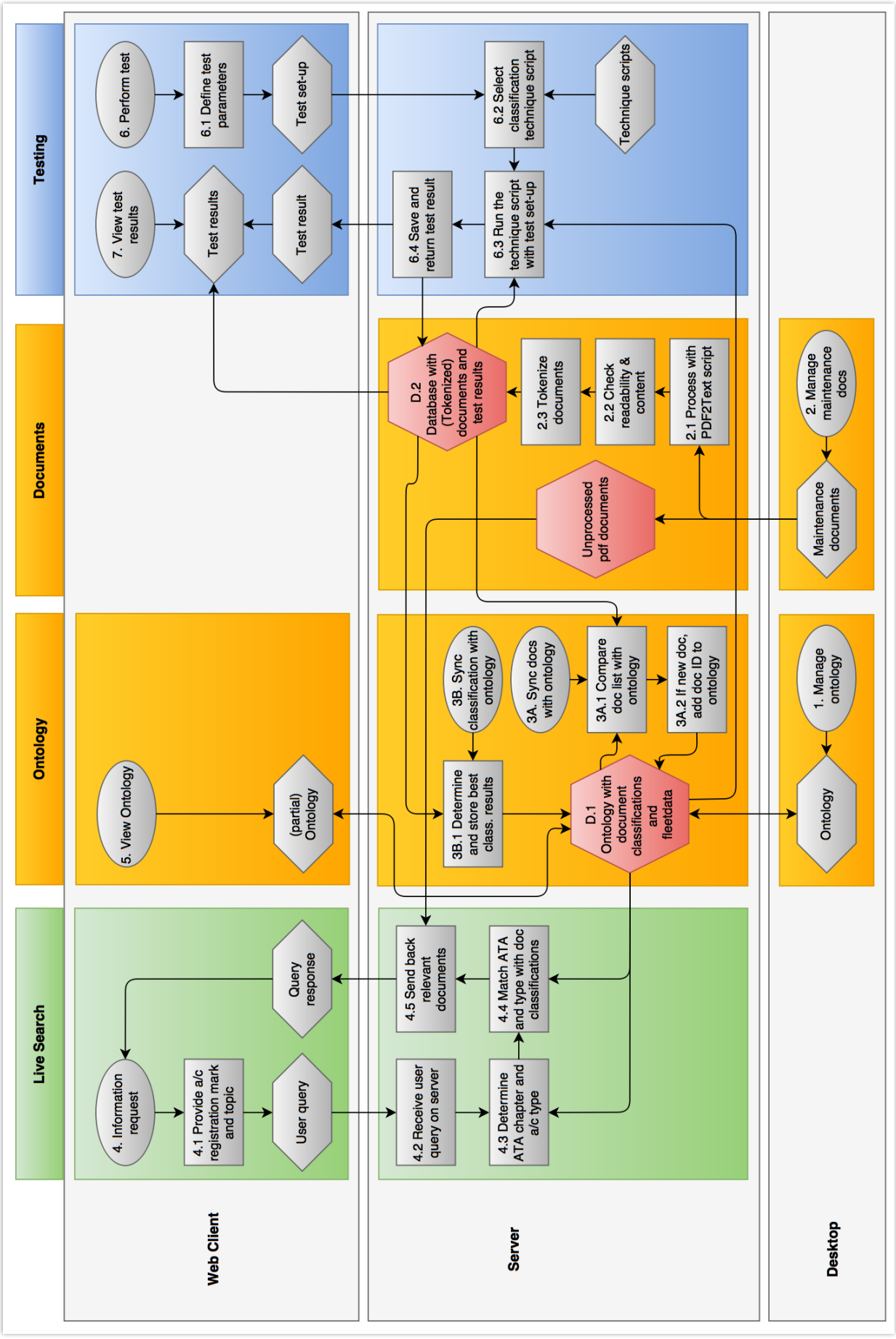


Figure 3.14: FFD of the Test Environment

3.3.2. System Functions

With the system components including hardware, software and data storage discussed, the next step is an overview of the functionalities incorporated in these components. Based on the theoretical FFD of chapter 2 and the top-level overview, Figure 3.14 shows an FFD of the functions developed on the server. It is a more detailed view of the overview as presented in Figure 3.1. It is structured in a matrix format, on the top row the 4 groups of functions are listed: Live Search, Ontology, Documents and Testing. On the first column, the different channels are given: the web client, server and desktop. In total 7 sets of functions are defined which are described below:

1. *Manage ontology* (desktop)
2. *Manage maintenance documents* (desktop)
3. *(A) Synchronize documents and (B) document classifications with ontology* (server)
4. *Information request* (client)
5. *View ontology* (client)
6. *Perform test* (client)
7. *View test results* (client)

Managing ontology (desktop)

The first set of functions is management of the ontology from the desktop. The functions are very simple, the ontology can be opened on the desktop with Protégé and stored, afterwards uploaded to the server (as an owl file) where it can be accessed by the server for processing.

Managing documents (desktop)

In a similar fashion to the ontology management, the document management is the interface between desktop and server for the documents. Documents can be uploaded and downloaded from the server by means of the FTP server and the server can process these documents. This is discussed in detail in section 3.2.

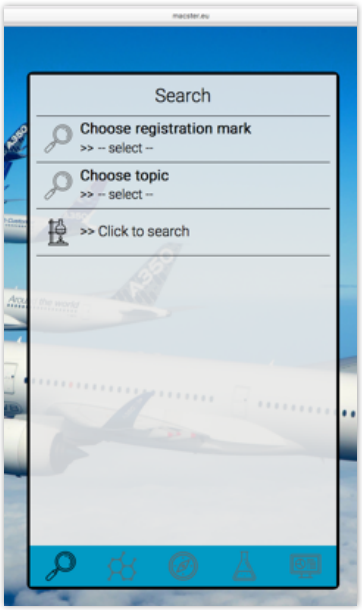


Figure 3.15: Live search page

Figure 3.16: Live search filled in

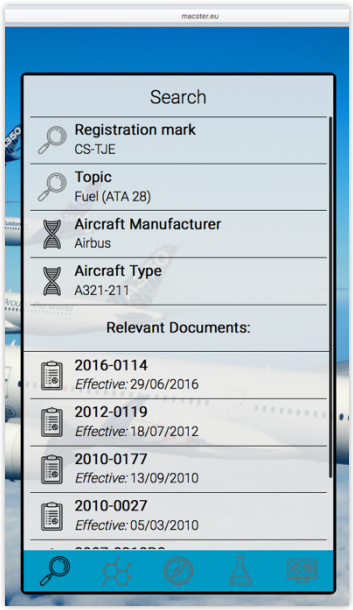


Figure 3.17: Live search results

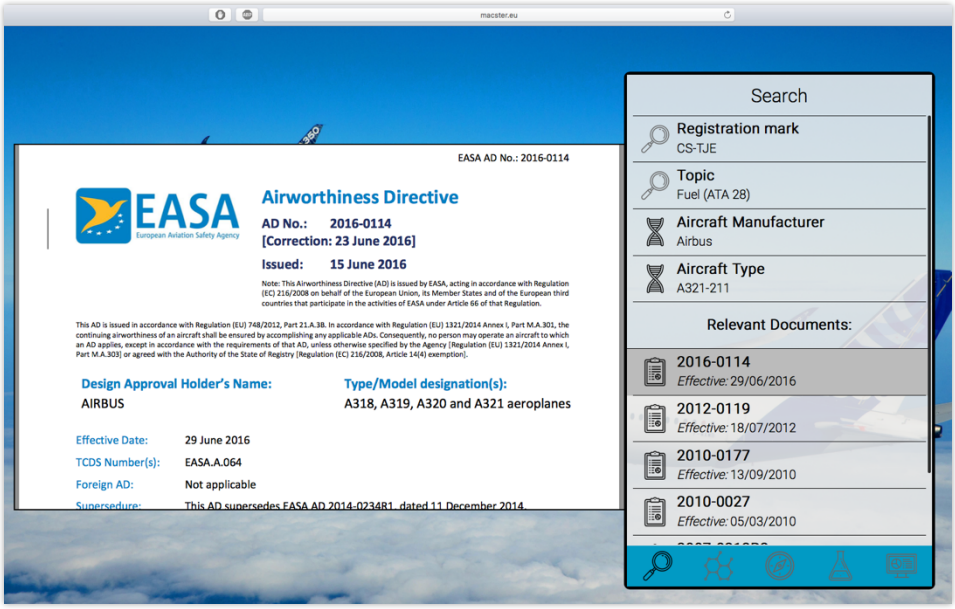


Figure 3.18: Viewing a document in live search

Syncing documents and document classifications with ontology (server)

There are two functions that are accessed on the server, which is syncing the documents and document classifications with the ontology. First, a script is written which checks all documents in the document database with the documents that are listed in the ontology (function 3A.1), if documents are added or removed, these documents are also updated in the ontology (function 3A.2). Updating the ontology means removing or adding the class from the ontology file, which is performed by the ontology interface script. The second functionality is syncing the document classifications with the ontology (function 3B). Based on the test results, the best classification techniques for each property are determined (function 3B.1) and the information obtained with these techniques is stored in the ontology so it can be used by the live search.

Live search (client)

The first part of the client is the live search, visualized in Figure 3.15, Figure 3.16, Figure 3.17 and Figure 3.18. Figure 3.15 shows the live search interface on the web client, with the two described inputs: Aircraft Registration Mark and Topic. After filling these in, of which an example can be seen in Figure 3.16, the search commences and after 1-2 seconds the results can be viewed as can be observed in Figure 3.17. The result is a list of relevant documents to the query, and in this overview the document name and effective date are given, these documents are also sorted by effective date. Additionally, the documents can be viewed from this interface, as shown in Figure 3.18. By just clicking on the document name, the document itself pops up to the left and can be viewed directly.

The functionality in more technical terms is provided in the FFD. Function 4.1 is entering the registration mark and the topic so that it can be send to the server (function 4.2). The server then matches the registration mark and topic with Aircraft Type and ATA chapter respectively (function 4.3). The Aircraft Type and registration mark are matched using the information in the fleet data of the ontology, whereas the ATA chapter and topic are linked via a list on the server (ATA chapter title and number). These two properties are then checked with the document list in the ontology and all ADs that match the hasDocumentAircraftType and hasDocumentATAChapter with the query are put in a list (function 4.4). This list is then checked on the properties hasDocumentCancellation, hasDocumentSuperseded and hasDocumentLanguage to match predefined requirements: no cancellation, not superseded and English respectively. The remaining documents that meet the

requirements are then obtained in pdf form from the unprocessed documents folder (function 4.5) and then send back to the user. The result is a list of documents with additional information such as the effective date at the user interface at the client. For these tests the validated data is used for all properties, so the information is not based on the techniques that are tested but on manually determined properties. An example of the results returned to the user can be seen in Figure 3.18.

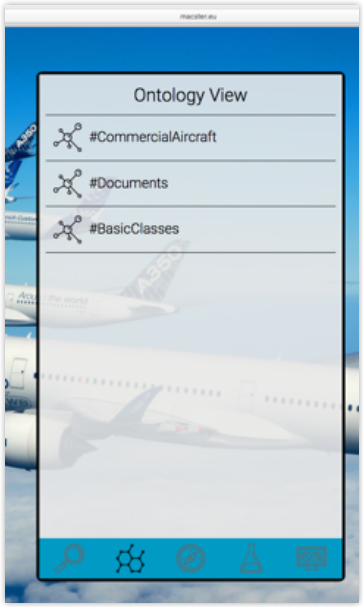


Figure 3.19: Ontology page

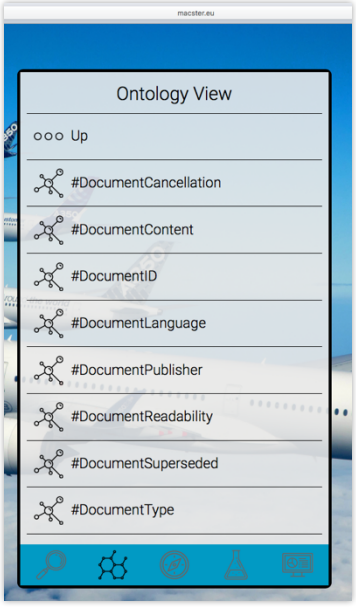


Figure 3.20: Ontology Aircraft Classes

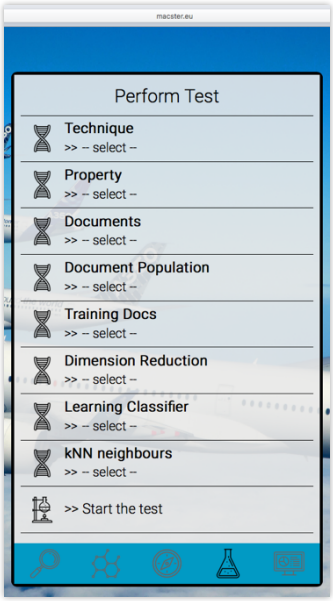


Figure 3.21: Perform test page

Viewing ontology (client)

The next element in the client is viewing the ontology. This top-level view is visualized in Figure 3.19, with the same groups Basic Classes, Documents and Commercial Aircraft as seen in Protégé. By clicking on the different groups, the sub-trees are visualized as can be seen in Figure 3.20. There is no new information versus the Protégé interface, but the main difference is that it is accessible via the client which simplifies the access, e.g. also on mobile devices. The functionality is quite simple; the ontology is requested by the web client and then loaded and provided by the server to the client.

Performing a test (client)

The last group of functions is related to the testing of techniques. An overview of the test page is given in Figure 3.21. The different parameters for the test can be seen, such as the technique, property and document population. More details about the test parameters is provided in section 3.4. The functionality as visualized in the FFD starts with a test set-up at the client (function 6.1), which is send to the server where after the right scripts are selected that match the technique provided (function 6.2). After selecting the right script, the test is performed (function 6.3). More details on the techniques is given in the next section (3.4) as well. When the test is completed, the results are stored in the database and send back to the user so they can be viewed (function 6.4).

Viewing test results (client)

As part of the 'Perform test' functions, the last step is viewing the test results. An overview of the test results page can be observed in Figure 3.22. It is a list of all the tests previously performed with a specific test id and time. Additionally, the details of every test can be seen by clicking on the test id, as visualized in Figure 3.23 (top part of the test result) and Figure 3.24 (bottom part of the same test result). The functionality to get the test results on the client is quite straightforward, they are requested and the server provides all previous test results.

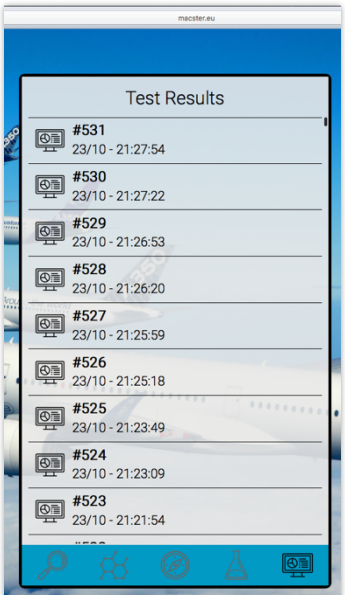


Figure 3.22: Test results overview

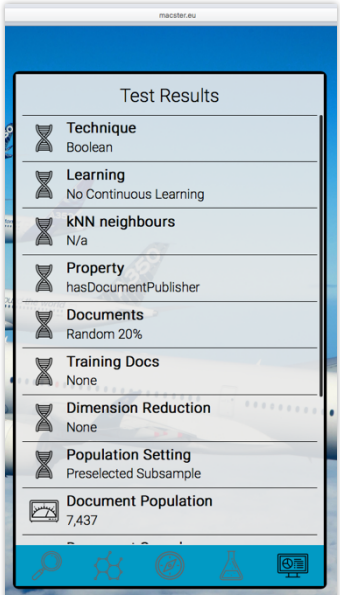


Figure 3.23: Test result (1)

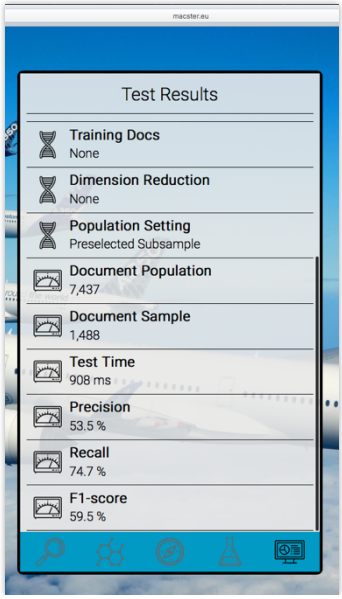


Figure 3.24: Test result (2)

3.4. Test Set-up

With the data selected and collected and the system overview provided in the last sections, the next step is to dive into the actual test set-up. The use case that is implemented in this system is to provide all ADs that match a given aircraft registration mark and topic. The aircraft registration mark is matched with an Aircraft Type by means of the fleet data in the ontology and the topic is linked with an ATA chapter on the server respectively. The aircraft type and ATA chapter must therefore be extracted from the ADs.

The ADs are not uniform documents, as they are available from different publishers (hence different templates), different languages, etc. The goal of the tests is to extract document properties that help in selecting the relevant subset of documents that can be used for the use case. As the TAP fleet is chosen, with only Airbus aircraft, the subset of documents that must be identified is English ADs that are applicable to Airbus aircraft. Determining each property is done using a selection of classification techniques from the previous chapter, which is elaborated later in this section. An example set-up is determining the class for the document property, either being English, French or German. The ‘correct’ class for all document properties is known beforehand by using metadata from the EASA tool, so the performance for each technique can be calculated using the predicted value by each technique and the actual value as it is known.

To get from the total population of 10571 ADs to this subset of English Airbus ADs, there are 3 properties that must be extracted from the documents: Document Publisher, Document Language and Document Aircraft Manufacturer. Two additional relevant properties for creating the right subset of relevant Airbus document are Document Type (to see if the document is an emergency AD or not) and Document Superseded (superseded documents are not relevant in the live search). From the final subset of documents, the Aircraft Type and ATA chapter can be extracted using the Boolean technique. These two properties cannot be tested with more advanced classification techniques as there are too few documents in the subset to perform a significant test.

The depicted components for the test set-up are described in detail below: the document properties, techniques, assessment criteria and a synthesis.

3.4.1. Properties

Document Publisher

The first property to extract from the documents is the document publisher. There are many possible document publishers, but 4 main publishers can be identified from the total population of 10571, as is visualized in Figure 3.25. The top-4 publishers are DGAC (3442 documents), which is the French government who published the French documents before EASA took over in 2003. It is followed by EASA (2986 documents), with only documents from 2003 onwards. The next two groups are FAA (2315 documents) and TC (869) which is Transports Canada. The other publishers combined are 959 documents and presented as 'Other' in the figure. This information is obtained using metadata provided by the EASA tool when downloading the documents.

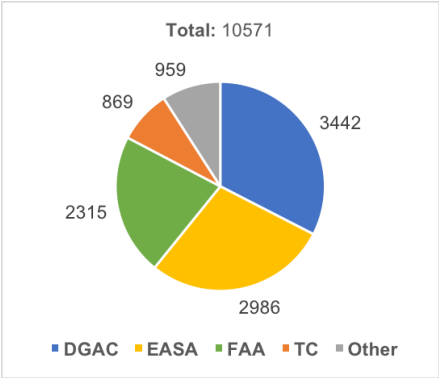


Figure 3.25: Document Analysis - Publisher

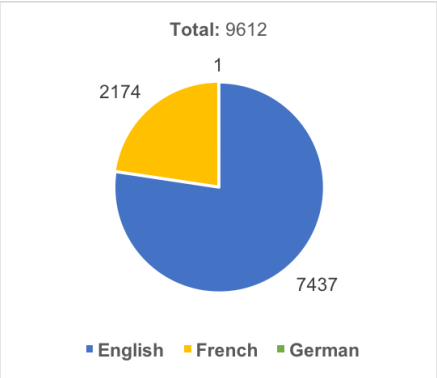


Figure 3.26: Document Analysis - Language

Document Language

The second property is the language used in the documents. For testing purposes, a subsample of the population is chosen from only the top-4 publishers. To perform a significant test, a large sample should be present and the languages of the documents from the 'Other' publishers are very different, hence resulting in very small sample sizes for each language. Removing the 959 documents from Other publishers from the total population results in a remaining sample of 9612 documents. There are 7437 English documents, 2174 French documents and 1 German document in this set, as can be seen in Figure 3.26. As with the document publisher property, this information is obtained by using metadata from the EASA tool and a small server script to divide the documents in the right language classes.

For testing purposes, a sample of 9611 is used instead of the 9612, as the 1 German document is removed from the test (too low a sample for that class). Additionally, for the live search capabilities of the system only English documents are selected, as the French documents are just translations of the English ones (or vice versa) and do not provide additional information.

Document Type

The third property to determine is the document type, which is either a regular Airworthiness Directive or Emergency Airworthiness Directive. The division between those two classes can be seen in Figure 3.27, there are 6976 regular ADs and 461 EADs in the entire population. An Emergency Airworthiness Directive is characterized by higher importance than a regular AD, but besides that there is no difference in the documents. This information is used to prioritize EADs over ADs in the search results of the system.

The test set-up for this property is slightly different from the previous properties, as another smaller document sample is made based on the two previous properties: only English documents from the top-4 publishers are selected. This is chosen due to the differences in documents of other publishers and the fact that French documents do not provide additional information over English ones. This results in a sub-population of 7437 documents that is sued for testing.

Document Superseded

The fourth property is Document Superseded, which is a property that defines whether an AD is superseded by a newer AD or not. Figure 3.28 shows the descriptive for this property: there are 5781 documents not superseded and 1656 ones are superseded. It is often clearly visible in an AD if it is superseded as a large red 'superseded' sign is then covering the entire document. The information whether a document is superseded is obtained using metadata from the AD website.

This property is used to remove superseded documents to show from the live search results, because superseded documents are not relevant to the user. The same sub-population of 7437 documents is used as for the document type, as the same logic for choosing this sample applies and it also means that the results for these properties can be compared.

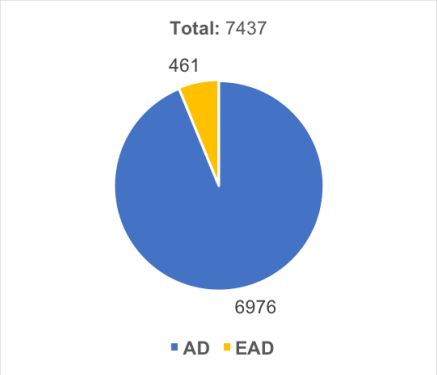


Figure 3.27: Document Analysis - Type

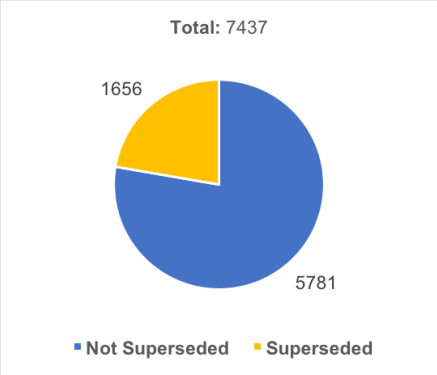


Figure 3.28: Document Analysis - Superseded

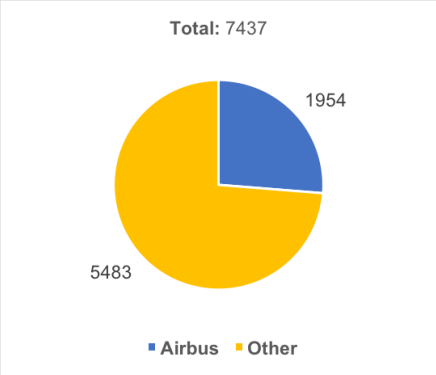


Figure 3.29: Document Analysis - Aircraft Manufacturer

Document Aircraft Manufacturer

The last property that is used for testing is the Document Aircraft Manufacturer, or in other words, the aircraft manufacturer a document is relevant to. For the specified use case, using the TAP fleet, a difference is made between Airbus and other aircraft manufacturers. Making this two-class distinction rather than a multi-class set-up with all manufacturers ensures that the sample sizes are large enough to perform relevant tests, otherwise some classes would only have a few documents which does not lead to significant results. As can be observed in Figure 3.29, there are 1954 Airbus documents and 5483 for other manufacturers. This information is again obtained from the EASA AD tool.

It is also the last property required to enable the search of the system, as Boolean methods are applied to extract ATA chapter and Aircraft Type from this subsample of relevant English Airbus documents.

3.4.2. Techniques

Now that the 5 properties to test are defined, the next step is selecting and setting up the techniques for these tests. As the tests are classification tests, only text classification techniques should be chosen with the simplest method being the Boolean technique. From the set of defined techniques in Chapter 2, there are 2 methods that can be applied to large texts: Naive Bayes and kNN. The other described methods in the literature are either very difficult to implement on large texts or too similar, for example in a basic set-up the SVM method works similarly to the kNN method. All three techniques did not have a library available in the npm database that could be used in the prototype system, therefore for each of these techniques a new script is developed. Below the three techniques are described as well as how these are implemented in the script.

Boolean

The Boolean technique is the simplest technique that is used in the testing. For each document property, a keyword is determined and compared with all document tokens (often words). If the keyword matches one of the tokens, the document matches the class that is assigned to that keyword. For example, in case of the property Document Publisher one of the keywords is 'FAA'. If FAA is one of the tokens in a document, the class 'FAA' is assigned to that document. This is then performed for all possibilities of classes, which in case of the publisher is DGAC, EASA, FAA, TC and Other (if it does not match any of the other classes). A similar procedure is performed for the other properties.

Naive Bayes

The first Machine Learning technique that is tested is the Naive Bayes technique. The Naive Bayes techniques requires a set of training documents to be able to determine the class of test documents. The document population for a property is first divided in a training sample and test sample. The first step of processing the training sample is to make a vocabulary per training class. For example, all training documents that are from the class English are combined in 1 list and likewise for all French documents. Based on these lists, a vocabulary is created per class. This vocabulary is a list of unique tokens and the frequency each word occurs in the training documents. For example, one of the tokens in the English list is 'balance' and the frequency is 7 in all training documents.

Having these vocabularies for both classes, the tokens in each test document can be compared with these vocabularies to determine the expected class of each test document. This comparison is performed in a probabilistic way and two main probabilities are used: the category probability and the probability for each token. The category probability is easily calculated by taking the amount of English documents / total documents in this example. The natural logarithm is taken from each probability to avoid underflow, which can happen when vocabularies grow over 100.000 tokens. The next step is calculating the probability for each token in the test document, which is performed in a similar fashion as in Chapter 2 but with Laplace Additive Smoothing. The Laplace smoothing is applied to account for the posterior probability, which is common in the Naive Bayes method. The token probability is then calculated by taking the natural logarithm of the wordFrequencyCount * (wordFrequencyCount + 1) / (class vocabulary size + total vocabulary size). The category probability and each token probability are then combined to get a total probability score for this class. The same calculations are then performed for the other classes of this property and finally the probabilities for both classes are combined: the lowest total probability is the expected class for the test document. The same procedure is then performed for the other test documents.

k-Nearest Neighbors

The second ML technique is kNN. It uses a similar starting approach as the Naive Bayes method, as it uses a training batch of documents to learn which word belongs to which class, and then for each test document all words are calculated with these training document words. The difference between the kNN method and the Naive Bayes method is that the kNN method calculates the Euclidian distance between the word frequency of each word in the test document and the training words and sums up the total distance per test document. Then the k 'closest' documents are selected and the class of these documents that has the highest value should be the class of the test document.

To start, the documents are again divided in a training set and a test set. The training documents and test documents are first both converted to a document vocabulary; a list per document of all unique words and their frequency (rather than 1 vocabulary for all documents of one class as with the Naive Bayes method). Each test document is compared to all training documents, by calculating the total Euclidian distance between each token. For each unique token in the test document, the frequency difference is calculated and squared. After the differences are all summed up, the square root is taken. After doing this comparison between the test document and all training documents, a list of total differences between the documents is obtained. Based on the selected 'k', the k documents that have the lowest distance are evaluated. The class of these documents are counted and the most counted class is the expected class for the test document. The same procedure can then be repeated for the other test documents.

3.4.3 Assessment Criteria

The previous subsection elaborated upon determining the expected classes of test documents and the next step is to evaluate these classes. Using the multi-class F-score calculation as described in Chapter 2, the F-score can be calculated for each test set-up.

3.4.4. Test set-up synthesis

The properties, techniques and assessment criteria are discussed, but there is still one element of the tests that should be elaborated: the documents. As the ADs have high randomness, there are many words that are not relevant to the given properties as these are aircraft / error specific, there is a bias in selecting a set of documents for each test. Therefore, to avoid this bias, for each test set-up (a combination of property and method), multiple tests are performed with a randomly picked test and training sample and the results are averaged. Unfortunately, as the precision and recall scores cannot be averaged for each test, only the F-score is used as a performance measure for the test. Additionally, a confidence interval of 95% the results is calculated (2σ) to check the range of the results of the 5 tests and the time required for each test is measured. This means that all parameters for the test are described and a standardized overview can be seen in Table 3.1.

Table 3.1: Test results set-up

Method	**Population**	**Sample**	**Training**	**Time (avg, s)**	**F_β (mean)**	**F_β (2σ)**
Document Property (options)						

3.5. Verification and Validation

Having established the test set-up an important element is the verification and validation of the system used for testing. In this section, both are elaborated upon.

3.5.1. Verification

The verification of the scripts related to the test set-up is to ensure that the tests are performed as expected, or in other words: if the techniques are correctly coded in the system. Additionally, the F-score calculation is verified. For the verification of the test scripts, the set-up is developed again in another tool: Microsoft Excel. All elements from tokenized documents to document classifications are coded as well in Excel for a specific use case, as only the functionality must be checked and not all results. The use case that is recreated is testing the language of all ADs from Transports Canada from the year 2010, while using all ADs of 2009 as a training set. This dataset is chosen as Transports Canada publishes all documents in an English and French version, which gives a convenient training and test set. There are 49 English and 49 French ADs in 2009, which are used for training. Furthermore, there are 39 English and 37 French ADs in 2010 (the offset is due to 2 extra English appendices). The verification procedure starts with the tokenized documents from these two samples.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	Doc	Language	Type	Tokens from here to right										
2	#EASA_AD_CF-2009-01R1_1.pdf	English	Training	Transport	Transports	Canada	Canada	TP	7245E	AIRWORTHINESS	DIRECTIVE	The	following	airworthiness
3	#EASA_AD_CF-2009-01R1_2.pdf	French	Training	Transports	Transport	Canada	Canada	TP	7245F	CONSIGNE	DE	NAVIGABILITvâ	La	prv©sente
4	#EASA_AD_CF-2009-02R1_1.pdf	French	Training	Transport	Transports	Canada	Canada	TP	7245F	CONSIGNE	DE	NAVIGABILITvâ	La	prv©sente
5	#EASA_AD_CF-2009-02R1_2.pdf	English	Training	Transport	Transports	Canada	Canada	TP	7245E	AIRWORTHINESS	DIRECTIVE	The	following	airworthiness
6	#EASA_AD_CF-2009-03_1.pdf	English	Training	Transport	Transports	Canada	Canada	TP	7245E	AIRWORTHINESS	DIRECTIVE	The	following	airworthiness
7	#EASA_AD_CF-2009-03_2.pdf	French	Training	Transports	Transport	Canada	Canada	TP	7245F	CONSIGNE	DE	NAVIGABILITvâ	La	prv©sente

Figure 3.30: Some tokenized documents in Excel

	A	B	C	D	E	F
1	document_name	consigne count	BOOLEAN test	Macster	Real	Verification
2	#EASA_AD_CF-2010-01_1.pdf	0	English	English	English	TRUE
3	#EASA_AD_CF-2010-01_2.pdf	20	French	French	French	TRUE
4	#EASA_AD_CF-2010-02_1.pdf	6	French	French	French	TRUE
5	#EASA_AD_CF-2010-02_2.pdf	0	English	English	English	TRUE
6	#EASA_AD_CF-2010-03_1.pdf	0	English	English	English	TRUE
7	#EASA_AD_CF-2010-03_2.pdf	6	French	French	French	TRUE
8	#EASA_AD_CF-2010-04_1.pdf	12	French	French	English	TRUE
9	#EASA_AD_CF-2010-04_2.pdf	0	English	English	French	TRUE
10	#EASA_AD_CF-2010-05_1.pdf	0	English	English	English	TRUE
11	#EASA_AD_CF-2010-05_2.pdf	26	French	French	French	TRUE
12	#EASA_AD_CF-2010-06R1_1.pdf	0	English	English	English	TRUE
13	#EASA_AD_CF-2010-06R1_2.pdf	6	French	French	French	TRUE
14	#EASA_AD_CF-2010-07_1.pdf	0	English	English	English	TRUE
15	#EASA_AD_CF-2010-07_2.pdf	0	English	English	French	TRUE
16	#EASA_AD_CF-2010-08_1.pdf	0	English	English	English	TRUE
17	#EASA_AD_CF-2010-08_2.pdf	16	French	French	French	TRUE
18	#EASA_AD_CF-2010-09_1.pdf	0	English	English	French	TRUE
19	#EASA_AD_CF-2010-09_2.pdf	8	French	French	English	TRUE
20	#EASA_AD_CF-2010-10_1.pdf	0	English	English	French	TRUE
21	#EASA_AD_CF-2010-10_2.pdf	10	French	French	English	TRUE

Figure 3.31: Part of the verification results for the Boolean method in Excel

Boolean

The verification of the Boolean technique is the easiest of the three techniques. The training data remains untouched and only the test data is processed. The tokenized test documents are loaded in Excel, of which an example can be viewed in Figure 3.30. The Boolean keyword to determine the language of the document is ‘consigne’, the French translation of directive. A simple IF statement is executed to check whether the word ‘consigne’ exists in one of the columns with document tokens and the result is either the French or English class. The result for each test document is compared

with the result that the system provided for each test document and can be observed in Figure 3.31. The result is a list of Excel results and system results and these are compared: all 76 test documents gave the same result for both methods, as expected.

Naive Bayes

A more complex verification process is required for the Naive Bayes technique. Using the 49 English and French training documents, a training vocabulary must first be created for both classes. To do this in Excel, all tokens from both sets of documents must be counted. The resulting numbers are verified along the way with the test environment. After creating the vocabularies, it is found that there are 38016 English tokens and 47919 French tokens. This is an interesting result as the documents are literal translations of each other and it was expected that these counts are more similar, instead of having a 26%-word count difference. Nevertheless, an example of the vocabulary is given in Figure 3.32. These counts are the same as in the test environment.

	A	B	C	D	E
1	English Words Name	English Words Count		French Words Name	French Words Count
1417	dielectric	3		contraintes	3
1418	Dielectric	1		contre	7
1419	differ	2		Controls	2
1420	different	1		convient	1
1421	difficulty	4		copilote	1
1422	digital	1		coque	1
1423	directional	2		corps	2
1424	DIRECTIVE	49		correctement	7
1425	directive	305		correction	1
1426	Directive	25		corrections	1
1427	directive)	1		corrective	7
1428	Directives	1		correctives	53
1429	directly	1		Correctives	1

Figure 3.32: Example of the verification vocabularies in Excel

Having established the training vocabularies, the documents can be tested. For each of the documents a vocabulary is created and for each of the tokens in the vocabulary the probability is calculated whether it matches the English or French documents, using the equations as described in the previous section. An example of these token probabilities for the first document of the test set is given in Figure 3.33. A more negative token probability means a lower likelihood a word matches that category compared to the other class. Combining the token probabilities for each class and adding the category probability, which is the natural logarithm of 0.5 as both training sets have equal number of documents, a total probability for this first test document is obtained. The total probability for the English class is -6349,25 and -9195,04 for the French class. In this case, the English class would be chosen by the technique as it is a higher probability than the French class. These total probabilities are identical to the results found by using the system. The same procedure was executed for the other test documents (the remaining 75 test documents) and these results were also identical, which verifies the Naive Bayes technique in the test environment.

	A	B	C	D	E	F	G
1	Words	Frequency in Text	English Word Frequency Count	French Word Frequency Count		English Token Probability (Laplace)	French Token Probability (Laplace)
214	non	2	13	41		-16,06	-14,28
215	conformity	1	3	0		-9,28	-10,88
216	increased	1	3	0		-9,28	-10,88
217	their	1	7	0		-8,59	-10,88
218	susceptibility	1	2	0		-9,57	-10,88
219	brittle	1	2	0		-9,57	-10,88
220	fracture	1	2	0		-9,57	-10,88
221	Subsequently	1	4	0		-9,06	-10,88
222	it	2	32	0		-14,35	-21,75
223	was	2	34	0		-14,23	-21,75
224	established	1	2	0		-9,57	-10,88

Figure 3.33: Naive Bayes token probability calculations for verification in Excel

	A	B	C	D
	Token	Word Frequency Test Doc	Word Frequency Training Doc	Eucl Dist
5	upon	1	1	0
7	compliance	4	2	4
8	with	10	6	16
9	all	2	1	1
0	Failure	2	1	1
1	comply	1	1	0
2	requirements	1	2	1
3	AD	2	8	36
4	invalidate	1	1	0
5	flight	4	1	9
6	authorization	1	1	0
7	Alternative	1	1	0
8	means	1	1	0
9	shall	2	2	0
0	applied	1	1	0
1	for	2	3	1
2	accordance	2	2	0

Figure 3.34: kNN verification Euclidian distance per word in Excel

	A	B	C
	Document	Total Distance	Language
	#EASA_AD_CF-2009-50_1.pdf	33,3166625	English
	#EASA_AD_CF-2009-48_1.pdf	40,23679908	English
	#EASA_AD_CF-2009-21R1_1.pdf	48,37354649	English
	#EASA_AD_CF-2009-08R1_1.pdf	49,25444142	English
	#EASA_AD_CF-2009-33_1.pdf	50,12983144	English
	#EASA_AD_CF-2009-20_1.pdf	50,19960159	English
	#EASA_AD_CF-2009-09_2.pdf	50,35871325	English
	#EASA_AD_CF-2009-05_1.pdf	50,45790325	English
	#EASA_AD_CF-2009-15_1.pdf	51,10772936	English
	#EASA_AD_CF-2009-19_1.pdf	51,41984053	English
	#EASA_AD_CF-2009-01R1_1.pdf	51,95190083	English
	#EASA_AD_CF-2009-47_2.pdf	52,26853738	English
	#EASA_AD_CF-2009-16_1.pdf	52,44044241	English
	#EASA_AD_CF-2009-25R1_1.pdf	52,64978632	English
	#EASA_AD_CF-2009-29R1_1.pdf	52,80151513	English
	#EASA_AD_CF-2009-46_1.pdf	53,46961754	English
	#EASA_AD_CF-2009-13_1.pdf	53,50700889	English
	#EASA_AD_CF-2009-22_1.pdf	53,94441584	English
	#EASA_AD_CF-2009-49R1_1.pdf	54,28627819	English

Figure 3.35: kNN verification Euclidian distance per document in Excel

k-Nearest Neighbors

In a similar fashion to the Naive Bayes technique, the kNN technique is verified. No class-based vocabulary is made as with the Naive Bayes method and the test documents can be compared immediately to the training documents. For this, the Euclidian distance is calculated for each token in the test document, of which an example can be observed in Figure 3.34. For each training document, the total Euclidian distance to the test document is summed up and ranked from low to high distance, as can be seen in Figure 3.35. It can be observed that for this test document, the class of all the closest documents is English, so the choice for k would not matter in this case. Nevertheless, this process is executed for all test documents in the system and the results are identical to the Excel method.

F-score calculation

Besides the verification of the techniques, the F-score calculation must also be verified. This is done using the results of the Boolean test. Based on the results, a confusion matrix can be made, which can be observed in Table 3.2.

Table 3.2: Verification Confusion Matrix

		Actual value	
		English	*French*
Predicted value	*English*	31 (TP)	8 (FP)
	French	7 (FN)	30 (TN)

From the confusion matrix 31 True Positives, 8 False Positives, 7 False Negatives and 30 True Negatives can be observed. Given these numbers, the precision, recall and F-score are calculated. The precision is 31 / (31+8) = 0.795 and the recall is 31 / (31+7) = 0.815. The corresponding F-score is (2*0.795+0.815) / (0.795+0.815) = 0.805 or 80.5%. The results obtained from the system on this specific test can be seen in Figure 3.36: 79.5%, 81.5% and 80.5% respectively.

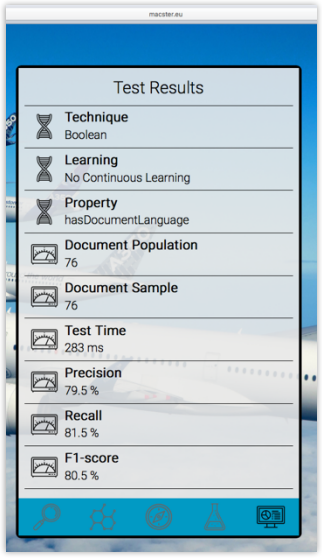


Figure 3.36: System F-score verification

3.5.2. Validation

To validate the system, the actual values for all properties that are tested should be checked: is the actual value correct or incorrect. For a valid system, the actual values as given from the metadata from EASA are correctly representing the document properties. For example, checking if the language of a document is English or French and comparing this with the actual value that is used in calculating the performance of the techniques. As it is impossible to manually check all 14.008 documents for all 5 properties (which gives 70.040 checks), 5% (or 700 documents) are randomly selected from the entire population, using a small validation script. These 700 documents are manually checked for all 5 properties that are used and it is found that all values were correct.

Additionally, the live search system is used to validate the results of the search. For each registration mark and topic combination (6x3), all documents returned are checked with the input query of the search by manually checking the Aircraft Type and ATA chapter in said documents. This second check also lead to the conclusion that the system is operating as required, so it can be concluded that the system is validated.

3.6. Pre-Test Sensitivity Analyses

The previous sections provided a description of the test environment, data and test set-up. Before the tests are performed to compare the performance of the techniques on the different properties, first a set of pre-tests is performed to determine how the tests can be set-up the best. There are three important parameters of the test that are determined this way: the amount of training data for the Naive Bayes and kNN method, the number of neighbors (k) for the kNN method and the sample size during the tests. These pre-tests are a sensitivity study to these parameters. This section is structured accordingly, the results for each of these parameters are discussed per section (3.6.1 – 3.6.3).

3.6.1. Training data

The first parameter for the sensitivity analysis is the amount of training data that is used by the methods based on Machine Learning, the Naive Bayes and kNN method. As these methods are reliant on training data, the first step is to determine how much training data is required for the best results, which is defined as the highest mean F-score. Additionally, the average processing time and 95% confidence interval are measured. As the training data is the most important parameter for the methods, the tests are performed on both the kNN and Naive Bayes method. For the kNN method a k=11 is chosen and kept constant for all tests, which is an arbitrary choice; section 3.6.2 elaborates on the choice of k. Furthermore, for both tests the property hasDocumentAircraftManufacturer is chosen, as there is enough data for both options (Airbus / Other) and it is a more complex task than for example hasDocumentLanguage.

The results for the kNN method are shown in Table 3.3. A total of 25 tests are performed, 5 for each set-up, which are then averaged. The method, population and sample are similar for all tests, however the documents in each sample are not as random samples are used. The training data used is 2*25 (25 Airbus and Other pre-classified documents), 2*50, 2*100, 2*250, 2*500. The results indicate an increasing F-score for increasing training size. The confidence interval for each set-up is up to 8.1%, however the increase in performance is substantially larger. Furthermore, the time required for each set-up increases with more training data used.

Table 3.3: Pre-Test Results for varying training data (kNN)

Method	Population	Sample	Training	Time (avg, s)	F_β (mean)	F_β (2σ)
hasDocumentAircraftManufacturer (Airbus / Other)						
kNN (11)	7,437	744	2*25	13.0	64.2%	8.1%
			2*50	23.4	68.0%	6.9%
			2*100	47.2	71.0%	5.4%
			2*250	121.8	77.0%	3.4%
			2*500	242.9	78.7%	3.6%

Similar results are obtained for the Naive Bayes method; these can be seen in Table 3.4. The method, population and sample are again the same for all tests, with the documents randomly selected in each sample. The training data increases similarly to the kNN method for each set-up. The results show the same performance as with the kNN method, for increasing training data the performance (F-score) also increases, however less than the kNN method. Additionally, the confidence interval is high for most set-ups. The main difference between the kNN and Naive Bayes tests is the time, which is substantially lower for the Naive Bayes method. Also, the difference between each Naive Bayes test set-up is negligible.

Table 3.4: Pre-Test Results for varying training data (Naive Bayes)

Method	Population	Sample	Training	Time (avg, s)	F_β (mean)	F_β (2σ)
hasDocumentAircraftManufacturer (Airbus / Other)						
Naive Bayes	7,437	744	2*25	2.0	62.7%	11.7%
			2*50	2.6	68.2%	11.3%
			2*100	2.4	67.8%	3.2%
			2*250	4.1	71.5%	7.7%
			2*500	3.8	73.2%	6.4%

The results for both the methods show a similar behavior and therefore it is concluded that the highest available amount of training data is the best set-up for each test using Machine Learning methods. The higher required time for the kNN method (in this case roughly 4 minutes per test) still is within limits and the time required for the Naive Bayes method is negligible.

3.6.2. k-Nearest Neighbors (k)

The second parameter is the number of neighbors (k) in the kNN method. The choice for k is very important for the performance of the technique but it is also impossible to determine beforehand as it is highly dependent on the task it is used for. This pre-test investigates the performance of five possible values for k: 3, 5, 7, 11, 19. Odd numbers are chosen as 4 out of 5 final tests have 2 classes each, so choosing an odd number always leads to a 'winner'. The value k=1 is not chosen due to the set-up with 2 classes each for most properties. The property hasDocumentAircraftManufacturer is chosen again and the population, sample size and training are kept constant for all tests, but the sample set is randomly chosen for each test. Only 100 training documents are chosen rather than the 500 determined in section 4.2, due to the time required for each test. In this pre-test, the highest possible F-score is not required, but the relative difference in F-score between each set-up. Again, 5 tests are performed with 5 different set-ups and the results are averaged per set-up.

Table 3.5: Pre-Test Results for varying k (kNN)

Method	Population	Sample	Training	Time (avg, s)	F_β (mean)	F_β (2σ)
hasDocumentAircraftManufacturer (Airbus / Other)						
kNN (3)	7,437	744	2*100	47.3	77.0%	5.5%
kNN (5)				46.2	75.4%	5.1%
kNN (7)				49.2	71.9%	12.0%
kNN (11)				47.2	71.0%	5.4%
kNN (19)				47.5	67.2%	7.3%

The results for this pre-test can be observed in Table 3.5. For increasing k, the mean F-score decreases. The decrease between 3 and 5 is minimal, but overall the trend is a decreasing performance., It should be noted that the confidence interval is rather large (especially for the case k=7), however this does not affect the downward trend. The time required for each test is very similar, so the choice of k does not influence the test time.

It can be concluded that the best choice for k is as small as possible, uneven and not k=1, which is k=3 in this case.

3.6.3. Sample size

The last parameter that influences the test set-ups is the sample size. As both training and sample documents are selected randomly from the population, variations in results exist. Therefore, varying the sample size can indicate an effect of this parameter on the actual outcome. For these tests, the method Naive Bayes is chosen on the property hasDocumentAircraftManufacturer, with a similar population as the previous pre-tests. A training set of 500 documents for each class is chosen and kept constant for all tests. Again, a total of 5 tests are performed on each of the different sample sizes: 5%, 10%, 20%, 50% and 100% of the population. This results in a sample size of 372, 744, 1488, 3719 and 6437 (which is not 7437 because there are 2*500 training documents).

The results of this pre-test are seen in Table 3.6 for each sample size. The F-score performance for each set-up is similar, with a range of 1.4%. This range is well within the 95% confidence interval for each set-up, so the results can be regarded as similar. Additionally, the time required to complete a test increases as the sample increases, as more documents must be processed.

Table 3.6: Pre-Test Results for varying sample size (Naive Bayes)

Method	Population	Sample	Training	Time (avg, s)	F_β (mean)	F_β (2σ)
hasDocumentAircraftManufacturer (Airbus / Other)						
Naive Bayes	7,437	372	2*500	2.2	74.6%	2.9%
		744		3.8	73.2%	6.4%
		1,488		6.4	74.6%	3.8%
		3,719		12.9	73.9%	3.8%
		6,437		25.0	74.0%	5.9%

From these results, it can be concluded that sample size does not influence the actual performance of each measure. Therefore, the sample size can be chosen using other requirements. The time required increases with a larger sample size, so a lower size than 100% can be chosen. The sample size of 20% (in this case 1488 documents) is chosen, as it is close to the training dataset size.

3.7. Conclusion

This chapter described the methodology used for the testing of the different techniques. First, the data is described and the selection of the Airworthiness Directivess as the data source. Subsequently, a system overview is given as well as a detailed description of all functions of this system. Lastly, the properties, techniques, assessment criteria and test set-up are given.

The pre-test sensitivity analysis showed that for the training data, it is concluded that more training data is better for the F-score performance. Therefore, in the test set-up the largest possible test data should be chosen. Furthermore, it is demonstrated that a small k (k=3) leads to the best results, so this is also chosen for the tests in the next chapter. Lastly, the sample size did not have a significant effect on performance in the test set-up made. Therefore, the sample size is chosen to be 20% of the population for the tests.

The next chapter provides the results of the tests with the given test set-up as well as a discussion of these results.

4. Test Results & Discussion

Chapter 4 shows the results of the tests on the properties defined in Chapter 3: Document Publisher, Document Language, Document Type, Document Superseded and Document Aircraft Manufacturer. Each test set-up is similar, as three methods are tested: Boolean, Naive Bayes and kNN (3). The results are presented and discussed per property in section 4.1. Afterwards, a synthesis and discussion of all findings is provided in section 4.2.

4.1. Tests Results for the five properties

4.1.1. Document Publisher

The first property to test is hasDocumentPublisher, which is extracting the document publisher from the Airworthiness Directivess. There are 5 classes for this property: DGAC, EASA, FAA, TransportsCanada and Other. The total population for this property is 10571, which is the entire available population. Furthermore, the sample size is 2115 documents (20% of total) and for the Machine Learning methods a training set of 500 documents per publisher is selected (so a total of 2500 documents). Three methods are tested: Boolean, Naive Bayes and kNN (3). The Boolean set-up for this property is searching on the terms 'secretarial' (the first word of the French publisher for this class), 'easa', 'faa' and 'transportscanada' respectively for each class, and if these are not found then the Other class is assigned. As with all test set-ups, 5 tests are performed for each method and the results are averaged.

Table 4.1: Test Results of Document Publisher

Method	Population	Sample	Training	Time (avg, s)	F_β (mean)	F_β (2σ)
hasDocumentPublisher (DGAC / EASA / FAA / TransportsCanada / Other)						
Simple Boolean	10,571	2,115	-	1.7	41.1%	0.4%
Naive Bayes			5*500	13.4	98.6%	0.4%
kNN (3)			5*500	1367.0	95.0%	1.1%

The results for this property can be seen in Table 4.1. The F-score of the Boolean method is 41.1% versus 98.6% for Naive Bayes and 95.0% for kNN (3). A significant performance gap can be identified between the Boolean method and the Machine Learning methods, of more than 50%. On the other hand, the Naive Bayes and kNN (3) method show relatively similar results at first glance. However, a difference in F-score of 4% is very significant for these methods, as literature demonstrated in chapter 2.

The confidence interval is relatively low for all methods, indicating steady performance over the 5 tests. However, the time required for each test is very different: the Boolean method is the quickest with 1.7s average test time, followed by the Naive Bayes method with 13.4s and lastly the kNN (3) method requires 1367.0s to complete.

The difference in results can be explained by looking at the type of property. The Document Publisher test consists of 5 different classes, or more accurately 4 classes and a 'rest' class. First, the Boolean method performs quite poorly as it only searches for 1 term in the documents. It was expected that this word was key to classify the documents, but the results indicate that this is not enough for a high F-score. On the other hand, the Machine Learning methods use the entire set of words in the document and are better able to capture the differences between document publishers. By looking at the documents from different publishers, each publisher uses its own template. This template consists of a certain amount of standard words in all documents of the publisher, which is recognized by the Naive Bayes and kNN method. The 'Other' class cannot be determined without knowing the document does not match the 4 main classes, as this class consists of documents that have many different templates (from all other publishers than the 4 largest), but still very high scores of 98.6% and 95.0% are obtained for the Naive Bayes and kNN (3) method respectively. Based on the literature, the difference between the Naive Bayes and kNN (3) method cannot be explained, so therefore more tests are performed to learn more about the differences in these methods.

An additional set of tests is performed on the property hasDocumentPublisher, by using the same population as for the other properties (except Document Language). In this smaller population, only documents from the publishers DGAC, EASA, FAA and TransportsCanada are chosen, with only the English language. This decreases the randomness in the documents, which theoretically improve the results of the Machine Learning methods. The training data is reduced to 250 documents per publisher, as there are only 440 documents from the TransportsCanada publisher and this is not enough to provide for both 500 training documents and sample. This test is performed to view the differences with the total population as well as comparing the results for this property with properties in next sections.

Table 4.2 shows the results of these tests. The mean F-score performance for the Boolean method increased to 58.6% compared to the previous test set-up, while the Naive Bayes and kNN (3) method stayed the same; 99.0% and 94.1% respectively. The confidence interval shows the same range as the previous test and the time required has a similar distribution between methods, but in general less due to the smaller training data. As the scores for the Machine Learning based methods were already high, there was less to improve for these methods. It is however an interesting finding that changing the population to an expected 'better' selected set (by eliminating the 'Other' category and only choosing the English documents) only marginally affect the F-score. This means that the Document Publisher can be used as a first property to test for when a new document is added to the population, and determine it being part of one of the 4 large publishers or another one.

Table 4.2: Test Results of Document Publisher (selected population)

Method	**Population**	**Sample**	**Training**	**Time (avg, s)**	**F_β (mean)**	**F_β (2σ)**
hasDocumentPublisher (DGAC / EASA / FAA / TransportsCanada)						
Simple Boolean	7,437	1,488	-	1.4s	58.6%	1.5%
Naive Bayes			4*250	7.7s	99.0%	0.2%
kNN (3)			4*250	465.7s	94.1%	1.8%

4.1.2. Document Language

The second property to test is document language, which uses a smaller population than the first document publisher test. In a similar fashion to the second document publisher test, only documents from DGAC, EASA, FAA and TransportsCanada are chosen and then only English and French documents remain, as well as 1 German document which is also removed from the population. This mean that 9611 documents remain in the population. This population is larger than the last document publisher test (7437 documents), because both English and French documents are present in this language test. The training data is again 500 pre-classified English and French documents and the sample size is 20%, a total of 1923 randomly chosen documents from the population. Five tests are performed for each of these methods and these results are averaged. The Boolean method is set-up by searching for the word 'consigne' in the documents, which is the French word for 'directive' and is expected to be present in all and only in French documents.

Table 4.3: Test Results of the property Document Language

Method	Population	Sample	Training	Time (avg, s)	F_β (mean)	F_β (2σ)
hasDocumentLanguage (English / French)						
Simple Boolean	9,611	1,923	-	1.6	79.2%	1.7%
Naive Bayes			2*500	7.0	100%	0.1%
kNN (3)			2*500	522.5	99.8%	0.2%

The results of the tests can be observed in Table 4.3. In terms of F-score performance, a score of 100% is obtained by the Naive Bayes test, with a close second the kNN (3) test with a score of 99.8%. In contrast with these high scores, the Boolean score is relatively low with 79.2%. Additionally, the confidence interval is very low for all tests. In terms of test time, the same order as previous tests can be seen, the Boolean test being the fastest with 1.6s time, then the Naive Bayes test with 7.0s and lastly the kNN (3) test with 522.5s.

The perfect score of the Naive Bayes method and the near-perfect one of the kNN (3) method can be explained by the way these methods work. They both use the entire set of words in each document and as there are many differences between French and English words, these methods can grasp these differences perfectly. As a large part of the documents consists of the words that define the class, they provide the best score. The Boolean method scores lower as apparently, the expected word 'consigne' was not in every document, still a high score of 79.2% is obtained. There is no other population or sample to test for this category as with the Document Publisher property.

4.1.3. Document Type

The third property to test is the document type. This property is an extra property to test and not crucial for selecting the right information in the prototype, but learning if a document is an Emergency AD or a regular AD can help in the information provision to the user of a system (Emergency given more priority for example). The test set-up for document type is comparable to what is seen before, the three methods and the population size of 7437 due to choosing the top-4 publishers and only English documents. Additionally, the training data is reduced to 250 as there are only 500 EADs in the set and training on 500 would result in no documents left for the test sample. Furthermore, the Boolean method uses the word 'emergency' for identifying the Emergency ADs.

Table 4.4: Test Results of the property Document Type

Method	Population	Sample	Training	Time (avg, s)	F_β (mean)	F_β (2σ)
hasDocumentType (AD / EAD)*						
Simple Boolean	7,437	1,488	-	1.4	97.1%	0.7%
Naive Bayes			2*250	4.7	45.3%	5.3%
kNN (3)			2*250	234.9	51.9%	3.4%

An overview of the results is presented in Table 4.4. A different set of scores is observed compared to the previous 3 test set-ups. First, the F-score for the Boolean method is near perfect with a score of 97.1%. Additionally, the Naive Bayes and kNN (3) scores are much lower, being 45.3% and 51.9% respectively. Furthermore, the respective confidence intervals are much higher for the Machine Learning based methods compared to the Boolean method. Between these learning methods, the kNN (3) method outperforms the Naive Bayes by more than 6% which is a significant difference in score. These scores were expected for this property, as there is nearly no difference between the Emergency ADs and regular ADs. Almost all words in the documents are exactly similar, with at least one key difference: Emergency ADs use the word emergency a few times. As these documents consists of hundreds if not thousands of words, the Naive Bayes and kNN (3) method are not able to grasp the difference as the number of irrelevant features far exceeds the few relevant features. The test time has a similar order as before, the Boolean method is the quickest with 1.4s on average, Naive Bayes performs in 4.7s and the kNN (3) method works in 234.9s.

4.1.4. Document Superseded

The fourth property is an important property in selecting relevant ADs vs irrelevant ADs: whether a document is superseded or not. Documents that are superseded are irrelevant to show to the user, so should be filtered out. In this test set-up, the same three methods are used on the same population as previous tests. The training data is again 500 documents as there are many Superseded and Not Superseded documents. The Boolean method uses the word ‘superseded’ to check whether a document is superseded or not.

Table 4.5: Test Results of the property Document Superseded

Method	Population	Sample	Training	Time (avg, s)	F_β (mean)	F_β (2σ)
hasDocumentSuperseded (Superseded / Not Superseded)						
Simple Boolean	7,437	1,488	-	1.0	50.5%	0.8%
Naive Bayes			2*500	6.4	63.3%	6.4%
kNN (3)			2*500	469.9	67.0%	3.6%

In Table 4.5 the test results can be viewed. This is the first property where the F-score of the kNN (3) method is the highest with 67.0%, compared to 63.3% for the Naive Bayes method and 50.5% for the Boolean method. The respective confidence intervals are 3.6%, 6.4% and 0.8%. As expected, the time required for a Boolean test is the lowest with 1.0s, then the Naive Bayes set-up with 6.4s and the kNN (3) requires the most time with 469.9s.

The results indicate that this is the most difficult property to extract from the documents, as the highest score is 67.0% versus 98+% from the main tests of the previous three properties. It was expected that the Boolean score would be higher than 50.5%, as not superseded documents would not have the word 'superseded' in it. Furthermore, the Naive Bayes and kNN (3) method are more able to grasp if a document is superseded or not, but by far not the scores as the previous tests. However, compared to tasks described in literature these scores are still moderately high.

4.1.5. Document Aircraft Manufacturer

The final property is the Document Aircraft Manufacturer, or in other words the Aircraft Manufacturer a document applies to. This property has two possible values in this test set-up, either being Airbus or Other. The Other class is a mixture of many manufacturers. For this property, the same set of methods, population, sample and training data is used as the previous tests. The Boolean method uses a similar strategy as before, by searching for 'Airbus'.

Table 4.6: Test Results of the property Document Aircraft Manufacturer

Method	Population	Sample	Training	Time (avg, s)	F_β (mean)	F_β (2σ)
hasDocumentAircraftManufacturer (Airbus / Other)						
Simple Boolean	7,437	1,488	-	1.0	93.1%	2.1%
Naive Bayes			2*500	6.4	74.6%	3.8%
kNN (3)			2*500	475.5	84.0%	2.5%

The results for this test set-up are presented in Table 4.6. It can be observed that the performance of each method is higher than the previous property. The Boolean method performs the best with 93.1%, followed by the kNN (3) method with 84.0% and the Naive Bayes method with 74.6%. The confidence intervals are relatively low for this property. The test times are 1.0s, 6.4s and 475s for the Boolean, Naive Bayes and kNN (3) methods respectively, which is known behavior from previous test set-ups.

These results demonstrate that this property is easier to identify in the documents than Document Superseded and Document Type, but not as easy as Document Publisher and Document Language. The difference between the highest and lowest score is not high, but the difference between the methods is significant. The kNN (3) method performs better than the Naive Bayes method, but is a slower method. The Boolean method is superior, because it focusses on the single word Airbus while the other methods focus on the entire bag of words.

4.2. Synthesis & Discussion

Having the results for 6 tests on 5 properties, these results can be synthesized and compared between properties. Almost all test set-ups have a method with a very high score (93%+), only Document Superseded scores lower with a score of 67%. This is a (positively) surprising result as lower scores for more complex tasks were expected based on literature. For all these properties except Superseded, the current techniques can identify and extract the relevant information almost perfectly.

Furthermore, looking at the type of properties a few findings can be synthesized. First, the Boolean method performs best on properties that are identified with a single word; this simplicity makes this technique also widely used. If this specific word to search for is known, it is highly accurate. Nevertheless, if it is unknown or based on multiple words, the performance decreases drastically.

For the properties that are identified by multiple words, such as Language or Publisher, the Machine Learning techniques perform significantly better. Especially for language, which is most of a document content, the Naive Bayes method scores 100% (and kNN (3) scores 99.8%). Very high scores and due to the nature of these methods the perfect property to extract. As a document from a publisher is characterized by a specific set of words (for example the use of British or American English), or the choice of words within the publisher's template, can be identified by these Machine Learning methods. The high randomness in the documents (the descriptions with many different aircraft parts in the documents) is not a reason for lower scores, which is a very positive result. The two tests on the Document Publisher property illustrated this, with nearly identical scores for the Machine Learning methods despite the removal of 'randomness' from the population.

Lastly, a difference between the Naive Bayes and kNN (3) method can be seen in these test results. The Naive Bayes scores better than the kNN (3) method for the properties Publisher and Language, while kNN (3) outperforms Naive Bayes for the properties Type, Superseded and Aircraft Manufacturer. By looking at the type of these properties, it can be concluded that the Naive Bayes method is better if more words in a document are relevant to the classification of a document. On the other hand, despite the relative scores being lower, the kNN (3) method performs better than the Naive Bayes method if there are less relevant words to the classification in the document. The Publisher and Language properties had many relevant words, while Type, Superseded and Aircraft Manufacturer had far less, so few that the Boolean method is the best method.

Table 4.7: Overview of all Test Results

	Simple Boolean	**Naive Bayes**	**kNN (3)**
Document Publisher	41.1%	98.6%	95.0%
Document Language	79.2%	100%	99.8%
Document Type	97.1%	45.3%	51.9%
Document Superseded	50.5%	63.3%	67.0%
Document Aircraft Manufacturer	93.1%	74.6%	84.0%

The best methods for each task are the following, as can be seen in Table 4.7:
- Publisher: 98.6% with Naive Bayes
- Language: 100% with Naive Bayes
- Type: 97.1% with Boolean
- Superseded: 67.0% with kNN (3)
- A/c Manufacturer: 93.1% with Boolean

These tests are performed on 'raw' methods, the results could be improved by changing the search terms for the Boolean method and by applying feature reduction methods to the kNN (3) and Naive Bayes methods, to have a bag of words per document with a higher number of relevant features compared to the current situation. This could increase the performance of the Machine Learning methods by a certain degree so that they can meet the Boolean scores on these properties.

5. Conclusions and Recommendations

This final chapter of this thesis describes the conclusions, research contributions, limitations and recommendations of this research.

5.1. Conclusions

From the perspective of maintenance execution efficiency and flight safety, the tradeoff an AMT must make between available time and searching for all information required during an unscheduled task is one that should be avoided. By creating a system that automatically extracts relevant information from maintenance documents so that these can quickly be provided in digital form to the maintenance technician, the maintenance process can be completed more efficiently. In this way, the risks induced by the tradeoff can be avoided. This thesis answered the following research question related to this issue:

What is the recommended approach to identify and extract relevant information from digital maintenance documentation to aid a maintenance technician during an unscheduled task?

It was found that in the literature of the aircraft maintenance domain little is available about systems that automatically identify and extract information from maintenance documents. Moreover, working with digital maintenance documentation rather than paper-based documents is already a rare occurrence: the switch from paper-based to digital documentation provisioning is not widely introduced in the aircraft maintenance domain yet. By looking at other domains that focus on techniques that can identify and extract information from generic documents, a functional flow diagram was established with the functionality of a system that can identify and extract information from documents. Using this FFD and the techniques available in these other domains, a prototype automated documentation system was created that could be used with any type of data. This prototype is then altered to work with the use case applied in this thesis: obtaining relevant information from ADs. Five document properties were extracted from these ADs using three classification techniques: Boolean, Naive Bayes and kNN.

For almost all document properties, a very high F-score (>93%) was obtained. It is concluded that the Boolean method performed best on properties that are captured with a single word in the text. However, if there is no single keyword describing the property, the performance decreases drastically. For the properties that consist of many words, it is found that the Machine Learning methods perform significantly better. Between those Machine Learning methods, the Naive Bayes technique showed better performance if a high amount of the words in a document are relevant to the classification of a document. On the other hand, the kNN technique performs better than Naive Bayes if the amount of relevant words to the classification in the document is lower. In other words, the kNN method is better in distinguishing relevant from irrelevant words in the text.

It can be concluded that for identifying and extracting information from ADs, the techniques to apply must be selected based on the property / task to get the best performance. In case of the ADs, this is using the Naive Bayes technique for the properties Publisher and Language, using the Boolean technique for Type and Aircraft Manufacturer and the kNN technique for the Superseded property. If another type of document would be used in this system, such as an Aircraft Maintenance Manual, the process of finding the best technique for each relevant property should be repeated. The results

in this thesis are therefore only applicable to Airworthiness Directives, but the systematic approach to find the best technique for each property can be used in other cases as well. It means that for each type of document and property in the documents, the system must be specifically designed and tested before a selection can be made.

Additionally, while the best performing techniques are chosen for each of the properties, it can also be concluded that this level of performance cannot be used yet in practice. A score of >93% for most properties is not high enough, as these scores mean that there are some missing relevant documents to a task. Before these techniques can be used in a fully working system, they should be further improved.

Lastly, it is concluded that a recommended approach for an automated documentation system can be made based on the functionalities of the generic prototype system designed in this thesis. For it to work with specific maintenance documents, such as ADs, it must be adjusted and techniques must be tested on each of the properties of such a maintenance document. By comparing the performance of each technique per property, the best approach can be found. For ADs, a combination of Boolean, Naive Bayes and kNN techniques provides the best performance in this approach.

5.2. Research Contributions

The research contributions are a combination of theoretical and practical findings described in this report. Each specific contribution is discussed below.

Functional overview of six generic research domains

The first contribution stems from the theoretical side of the project, which is the creation of a functional overview of six generic research domains: NLP, IR, IE, TM, ML and KDT. The functionalities from each of these domains are combined into one novel overview, which covers document processing, ad-hoc processing and pre-processing of documents. This functional overview can be used for any type of documents and is therefore a research contribution to these six generic domains. Additionally, the functional flow diagram can be applied within the aircraft maintenance domain by applying a thesaurus or dictionary with information from the domain.

Extendable prototype automated documentation system

A major practical research contribution is the creation of a prototype automated documentation system. This system is used in this thesis to test different techniques on properties from ADs. However, the basics of the system can be applied in any context, e.g. it can be used to find information in research papers. In that case, most the system can remain intact and only specific user interfaces must be altered: using other search queries, changing the test setup options, using another dataset and the application of the techniques on that dataset must be changed. The specific novel areas of the prototype system are discussed below.

- #### Development of two Machine Learning scripts

  As an element of the prototype system, scripts were developed for two Machine Learning techniques: Naive Bayes and kNN. No existing libraries were available for these two techniques that could be applied directly for this case, so scripts were written to alter these basic techniques to work in the prototype system and with the AD data.

- ### Client user interface for testing different techniques
  A modular page is created where all elements for a test setup can be entered. The setup options can be easily changed and more options can be added, which enables the use of the automated documentation system in other cases as well.

- ### Live search user interface
  The system also has a live search user interface, which enables the retrieval of relevant documents based on a user query. In this thesis, only the aircraft registration mark and a topic were used for the query, but the system is designed so that these queries can be altered.

- ### Ontology and JavaScript interface
  No library was available that enabled an interface between an OWL/XML ontology and JavaScript, a novel interface is developed that can load an ontology and store the information in an array for further processing. Furthermore, new information can be immediately added to this array. The array can then also be saved again as an OWL/XML ontology file so it can be exported to for example a desktop and opened in Protégé.

- ### Interfaces between all system components
  By connecting all user interfaces of the automated system, the information obtained from for example the testing of properties can immediately be used in the live search user interface and accessed in the ontology. In practice, this could mean that when an information update is necessary in the system, it can be sent immediately to all users of the system. With the system 24/7 online, this literally means an immediate information update.

Database of ADs with many properties for testing techniques
A third contribution of this research is the creation of a database of ADs. This database is created in the ontology, which enables linking of the ADs with many properties. The database is also created so that it can be extended for other types of documents and more properties can be added easily. Most importantly, the database consists of 14.008 AD documents and up to 15 properties per AD. For this thesis, only a subset of these properties is used for testing. The database can therefore be used in further research to test on e.g. other properties and documents.

Comparison of techniques in the generic and aircraft maintenance domains
Lastly, the comparison of techniques made in this thesis is a research contribution. In the generic domains, this comparison underlines the findings from the literature in Chapter 2. The techniques show different results when it comes to different tasks (or properties in this case). By selecting the best technique for a specific task, a system can be created that offers high performance on multiple properties. The comparison is executed using data from the aircraft maintenance domain: The ADs. Therefore, this research also provides a novel comparison between techniques in the literature in this area, as such a comparison was not yet available.

5.3. Limitations

The system used in this thesis is generic and can be used for many applications, but it is designed for a very specific scope. The results of the tests are only valid for Airworthiness Directives, the five defined properties and the three techniques. The conclusions that can be drawn related to other maintenance documents is that a system should be specifically designed for the task to get a high performance. Furthermore, only English and French documents were used in the tests and this means that the results are also limited to these documents. It is expected that results would be comparable for other languages as well, but this is not tested in this thesis.

Furthermore, only Airbus aircraft are used in the prototype system as the TAP fleet is chosen. The selected relevant documents are therefore only applicable for Airbus aircraft. The system could however also be developed for another fleet, including aircraft form other manufacturers such as Boeing.

Additionally, the documents used in this system must be in a pdf format to be read by the system. This script is based on PDF2Text, but it can be extended for use with other file formats. Furthermore, older pdf documents could not be read, which led to 2298 of the 14008 documents not being readable. The older documents are however also less used in practice, as the aircraft to which they apply are often not in the air anymore.

5.4. Recommendations

A recommendation for future work in terms of the theoretical part of this thesis can be made related to the domains used. Instead of just selecting three major specific domains (medical science, biology, manufacturing) more literature from specific domains can be researched. In those other domains, relevant approaches, methods and techniques might be available, as well as better performance measures in more controlled environments. Additionally, the created functional flow diagram provides a generic overview for identification and extraction of relevant information from documents. It could be further extended by creating a diagram for the aircraft maintenance domain by using specific knowledge for this domain, e.g. a taxonomy, dictionary or ontology.

As this thesis provides a demonstration of a system that could automatically classify ADs, there are many steps that can be further extended in future work. First, only three techniques were tested in this thesis. This could be improved to include more of the techniques that are described in the theory chapter. To enable the testing of more techniques, it is necessary to use so-called feature extraction / feature reduction methods that decrease the number of features (words) drastically in documents, after which also techniques can be applied that rely on a small number of features per document such as the decision tree classification method. Furthermore, applying such feature reduction methods can also result in improved scores for the current techniques tested. For example, by reducing the amount of random words (that have a low frequency in all documents), the performance of the Machine Learning techniques may be improved. The effect of using a domain-specific dictionary or taxonomy could also be tested on the performance of techniques.

In this thesis, five properties are included, but more properties can be used for testing. Other relevant properties that could be extracted are the issue date, effective date, ATA chapter, aircraft type and references to other documents. These properties are more difficult to test, as many classes are possible per property, which thus requires a very large sample size to obtain significant results. Nevertheless, much more relevant information can be obtained from these documents.

Additionally, in this thesis only ADs are considered. Using this dataset provided a solid first step, but more document types such as Service Bulletins or the Aircraft Maintenance Manual could be incorporated in the prototype system. These documents have other structures and other relevant properties, so a different test setup would be required when considering these types of documents.

The testing of five properties was performed on specific populations and samples, by selecting specific classes of properties to ensure a good sample for testing. However, this selection of classes could also be incorporated into the testing. By using the fact that a document is issued by the Document Publisher FAA, an estimate can be used by specific techniques (such as the Naive Bayes) for the likelihood of English for the property Document Language. Using the information given in each property, and combining results from different properties, the overall results could be even further improved. The other way around is also possible, by combining multiple algorithms on a single property. For example, the Naive Bayes is best capable to work with entire documents, while techniques that are not used such as decision trees are best applied with few features. Using the Naive Bayes technique first to extract a few features from the documents which are then used by the decision tree method for further processing, an improved F-score could be obtained for the given five properties or even other properties. In a similar fashion, only parts of the document (sections) could be selected on which the techniques are applied.

Another major direction that can be further developed is the creation of an interface for the aircraft maintenance technician. In this thesis, a simple search tool is created from which a list of relevant ADs can be obtained and the contents of those ADs can be viewed. It is created with the AMT in mind, but the tool was not tested by AMTs themselves as that was not within the scope of this thesis. However, as the system is intended for the AMTs, the Human Machine Interface aspects of the system can also be further researched.

Bibliography

Afantenos, S., Karkaletsis, V. & Stamatopoulos, P., 2005. Summarization from medical documents: A survey. *Artificial Intelligence in Medicine*, 33(2), pp.157–177.

Airbus, 2016. E-Solutions. Available at: http://www.airbus.com/support/maintenance-engineering/e-solutions/ [Accessed February 1, 2016].

Aljumaili, M. et al., 2012. Study of aspects of data quality in e-maintenance. *International Journal of Condition Monitoring and Diagnostic Engineering Management*, 15(4), p.3.

Amardeilh, F. et al., 2013. Semi-automatic ontology maintenance in the virtuoso news monitoring system. *Proceedings - 2013 European Intelligence and Security Informatics Conference, EISIC 2013*, pp.135–138.

Ando, R.K. & Tong, Z. (Yahoo R., 2005. A Framework for Learning Predictive Structures from Multiple Tasks and Unlabeled Data. *Journal of Machine Learning Research*, 6, pp.1817–1853.

Appelt, D.E. et al., 1995. SRI International: Description of the FASTUS System Used for MUC-6. In *Proceedings of the Sixth Message Understanding Conference (MUC-6)*. pp. 237–248.

Atak, A. & Kingma, S., 2011. Safety culture in an aircraft maintenance organisation: A view from the inside. *Safety Science*, 49(2), pp.268–278.

Bangemann, T. et al., 2004. PROTEUS-An integration platform for distributed maintenance systems. In *4th International Conference on Intelligent Maintenance Systems-IMS'2004*. p. 9--p.

Batet, M., Sanchez, D. & Valls, A., 2011. An ontology-based measure to compute semantic similarity in biomedicine. *Journal of Biomedical Informatics*, 44(1), pp.118–125.

Bird, S., Klein, E. & Loper, E., 2009. *Natural language processing with Python*, " O'Reilly Media, Inc."

Brahma, M. et al., 2006. TEMIC: a New Cooperative Platform for Industrial Tele-Maintenance. In *The 2nd International Conference on Distributed Frameworks for Multimedia Applications*. IEEE, pp. 1–9.

Brusic, V. & Zeleznikow, J., 1999. Knowledge discovery and data mining in biological databases. *The Knowledge Engineering Review*, 14(3), pp.257–277.

Cambria, E. & Hussain, A., 2012. *Sentic computing: Techniques, tools, and applications*, Springer Science & Business Media.

Cambria, E. & White, B., 2014. Jumping NLP Curves: A Review of Natural Language Processing Research [Review Article]. *IEEE Computational Intelligence Magazine*, 9(2), pp.48–57.

Candell, O., Karim, R. & Söderholm, P., 2009. eMaintenance-Information logistics for maintenance support. *Robotics and Computer-Integrated Manufacturing*, 25(6), pp.937–944.

Caramia, M. & Felici, G., 2006. Mining relevant information on the Web: a clique-based approach. *International Journal of Production Research*, 44(14), pp.2771–2787.

Carbonell, J.G., Michalski, R.S. & Mitchell, T.M., 1983. Machine learning: A historical and methodological analysis. *AI Magazine*, 4(3), p.69.

Choudhary, A.K., Harding, J.A. & Tiwari, M.K., 2009. Data mining in manufacturing: A review based on the kind of knowledge. *Journal of Intelligent Manufacturing*, 20(5), pp.501–521.

Civil Aviation Authority, 2002. *Human factors in aircraft maintenance and inspection [CAP 718]*,

Cleverdon, C., 1960. Report on the first stage of an investigation into the comparative efficiency of indexing systems (ASLIB Cranfield Research Project). *Cranfield: College of Aeronautics, Sept., 166p*.

Collobert, R. et al., 2011. Natural Language Processing (almost) from Scratch. *Journal of Machine Learning Research*, 1, pp.1–48.

Cordón, O. et al., 2003. A review on the application of evolutionary computation to information

retrieval. *International Journal of Approximate Reasoning*, 34(2–3), pp.241–264.

Cowie, J., Lehnert, W. & Wilks, Y., 1996. Information extraction. *Communications of the ACM*, 39(1), pp.80–91.

Craven, M. & Kumlien, J., 1999. Constructing biological knowledge bases by extracting information from text sources. *Proceedings of the International Conference on Intelligent Systems for Molecular Biology (ISMB-99)*, pp.77–86.

Crespo, F. & Weber, R., 2005. A methodology for dynamic data mining based on fuzzy clustering. *Fuzzy Sets and Systems*, 150(2), pp.267–284.

Cunningham, S.J., Witten, I.H. & Littin, J., 1999. Applications of machine learning in information retrieval. *Annual Review of Information Science*, 34, pp.341–384.

Davis, J. & Goadrich, M., 2006. The Relationship Between Precision-Recall and {ROC} Curves. *International Conference on Machine Learning {(ICML)}*.

Dechamp, L. & PROTEUS WP2 Team, 2004. On the Use of Artificial Intelligence for Prognosis and Diagnosis in the PROTEUS E-maintenance platform. *In: Proceedings of MECHROB'04 — international conference on mechatronics and robotics, Aachen, Germany*.

Deerwester, S., Dumais, S.T. & Harshman, R., 1990. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6), pp.391–407.

Doan, S. et al., 2010. Integrating existing natural language processing tools for medication extraction from discharge summaries. *Journal of the American Medical Informatics Association : JAMIA*, 17(5), pp.528–31.

Dubitzky, W. & Azuaje, F., 2004. *Artificial intelligence methods and tools for systems biology*,

Esposito, F., Malerba, D. & Semeraro, G., 1995. A knowledge-based approach to the layout analysis. *Proceedings of 3rd International Conference on Document Analysis and Recognition*, 1, pp.466–471.

Farley, B., 2001. Extracting information from free-text aircraft repair notes. *AI Edam*, 15(4), pp.295–305.

Farley, B., 1999. From free-text repair action messages to automated case generation. *Proceedings of AAAI 1999 Spring Symposium*.

Fayyad, U., Piatetsky-shapiro, G. & Smyth, P., 1996. From data mining to knowledge discovery in databases. *Advances in Knowledge Discovery and Data Mining*, 17(3), pp.1–36.

Feldman, R. & Dagan, I., 1995. Knowledge Discovery in Textual Databases (KDT). *International Conference on Knowledge Discovery and Data Mining (Kdd)*, pp.112–117.

Feldman, R., Dagan, I. & Hirsh, H., 1998. Mining text using keyword distributions. *Journal of Intelligent Information Systems*, 10(3), pp.281–300.

Feldman, R. & Sanger, J., 2007. *The Text Mining Handbook*,

Fernández, M. et al., 2011. Semantically enhanced Information Retrieval: An ontology-based approach. *Web Semantics: Science, Services and Agents on the World Wide Web*, 9(4), pp.434–452.

Frasconi, P., Soda, G. & Vullo, A., 2002. Hidden Markov Chains for text categorization in multi-page documents. *Journal of Intelligent Information Systems*, 18(2/3), pp.195–217.

Frawley, W.J., Piatetsky-shapiro, G. & Matheus, C.J., 1992. Knowledge Discovery in Databases : An Overview. *AI Magazine*, 13(3), pp.57–70.

Fuhr, N. & Buckley, C., 1991. A probabilistic learning approach for document indexing. *ACM Transactions on Information Systems*, 9(3), pp.223–248.

Fukumoto, F., Suzukit, Y. & Fukumoto, J., 1997. An automatic extraction of key paragraphs based on context dependency. In *Proceedings of the fifth conference on Applied natural language processing*. Morristown, NJ, USA: Association for Computational Linguistics, pp. 291–298.

Garcia, E. et al., 2004. A new industrial cooperative tele-maintenance platform. *Computers and Industrial Engineering*, 46(4 SPEC. ISS.), pp.851–864.

Garg, A. & Deshmukh, S.G., 2006. Maintenance management: literature review and directions. *Journal of Quality in Maintenance Engineering*, 12(3), pp.205–238.

Gargiulo, F. et al., 2014. Aerospace Information System based on Semantic Technonogies and Ontology Management - A Web Portal for Semantic Search and Document Categorization. *Proceedings of 3rd International Conference on Data Management Technologies and Applications*, pp.341–348.

Gunjal, S.N., 2014. A Novel Ontology based R & D Project Proposal Classification using Text Mining Approach. *International Journal of Computer Applications*, 108(4), pp.23–28.

Hakenberg, J. et al., 2004. Finding kinetic parameters using text mining. *Omics : a journal of integrative biology*, 8(2), pp.131–152.

Hao, Y. et al., 2005. Discovering patterns to extract protein-protein interactions from the literature: Part II. *Bioinformatics*, 21(15), pp.3294–3300.

Harding, J. a. et al., 2006. Data Mining in Manufacturing: A Review. *Journal of Manufacturing Science and Engineering*, 128(4), p.969.

Hatzivassiloglou, V., Duboué, P. a & Rzhetsky, a, 2001. Disambiguating proteins, genes, and RNA in text: a machine learning approach. *Bioinformatics (Oxford, England)*, 17 Suppl 1, pp.S97–S106.

Hearst, M. a., 1999. Untangling text data mining. *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics on Computational Linguistics*, pp.3–10.

Hendler, J.A., 1996. Intelligent Agents: Where AI Meets Information Technology [Guest Editorial]. *IEEE Expert*, 11(6), p.20.

Himmelmann, N.P., 1998. Documentary and descriptive linguistics. *Linguistics*, 36(1), pp.1–34.

Hogan, R., Cesarone, T. & Dragun, D., 2003. Battle group automated maintenance environment. In *Proceedings of the 13th annual international ship control systems symposium explores automation in ship control, Philadelphia, USA*.

Holmberg, K., 2005. Maintenance, Condition Monitoring and Diagnostics. *In: POHTO 2005 International seminar on maintenance, condition monitoring and diagnostics, Oulu, Finland, 2005.*, pp.1–14.

Hu, Q., Huang, J.X. & Hu, X., 2012. Modeling and mining term association for improving biomedical information retrieval performance. *BMC bioinformatics*, 13(Suppl 9), p.S2.

IATA, 2013. *IATA's initiative on: Paperless Aircraft Operations*, Available at: https://www.iata.org/whatwedo/ops-infra/Pages/paperless-ops.aspx.

Jagannathan, V. et al., 2008. Assessment of commercial NLP engines for medication information extraction from dictated clinical notes. *International Journal of Medical Informatics*, 8, pp.284–291.

Jain, A.K., 2010. Data clustering: 50 years beyond K-means. *Pattern Recognition Letters*, 31(8), pp.651–666.

Jain, A.K. et al., 2000. Data Clustering: A Review.

Jantunen, E. & Gilabert, E., 2010. e-Maintenance: a means to high overall efficiency. *Engineering Asset Lifecycle Management*, (September), pp.688–696.

Jardine, A.K.S., Lin, D. & Banjevic, D., 2006. A review on machinery diagnostics and prognostics implementing condition-based maintenance. *Mechanical Systems and Signal Processing*, 20(7), pp.1483–1510.

Jiang, M. et al., 2011. A study of machine-learning-based approaches to extract clinical entities and their assertions from discharge summaries. *Journal of the American Medical Informatics Association*, 18(5), pp.601–606.

Karanikas, H. & Theodoulidis, B., 2002. Knowledge discovery in text and text mining software. *Centre for Research in Information Management, Department of Computation*.

Kaufman, M., 1998. Proceedings of the Seventh Message Understanding Conference (MUC-7).

Kelly, D. & Sugimoto, C.R., 2013. A systematic Review of Interactive Information Retrieval Evaluation Studies 1967- 2006. *International Review of Research in Open and Distance Learning*, 14(4), pp.90–103.

Kinnison, H.A. & Siddiqui, T., 2012. *Aviation maintenance management*,

Lampe, M., Strassner, M. & Fleisch, E., 2004. A ubiquitous computing environment for aircraft maintenance. *Applied Computing 2004 - Proceedings of the 2004 ACM Symposium on Applied Computing*, 2, pp.1586–1592.

Langley, P. & Simon, H. a., 1995. Applications of machine learning and rule induction. *Communications of the ACM*, 38(11), pp.54–64.

Lee, S.G. et al., 2008. Product lifecycle management in aviation maintenance, repair and overhaul. *Computers in Industry*, 59(2–3), pp.296–303.

Léger, J.B., 2004. A case study of remote diagnosis and e-maintenance information system. Keynote speech of IMS'2004. In *International conference on intelligent maintenance systems, Arles, France*.

Levrat, E. & Iung, B., 2007. TELMA: A full e-maintenance platform. *Second World Congress on Engineering Asset Management and the Fourth International Con- ference on Condition Monitoring, WCEAM/CM 2007*.

Levrat, E., Iung, B. & Crespo Marquez, a., 2008. E-maintenance: review and conceptual framework. *Production Planning & Control*, 19(4), pp.408–429.

Lewis, D.D., 1991. Learning in intelligent information retrieval. *Machine Learning: Proceedings of the Eighth International Workshop*, pp.235–239.

Liao, S.H., Chu, P.H. & Hsiao, P.Y., 2012. Data mining techniques and applications - A decade review from 2000 to 2011. *Expert Systems with Applications*, 39(12), pp.11303–11311.

Lufthansa Technik AG, 2016. Practically paperless. Available at: http://www.lufthansa-technik.com/paperless-mtc [Accessed February 1, 2016].

Malin, J.T. & Throop, D.R., 2007. Basic concepts and distinctions for an aerospace ontology of functions, entities and problems. *IEEE Aerospace Conference Proceedings*.

Manning, C.D., Raghavan, P. & Schutze, H., 2009. *An Introduction to Information Retrieval*, Cambridge, England: Cambridge University Press.

Manning, C.D. & Schütze, H., 1999. *Foundations of statistical natural language processing*, MIT Press.

Martin, J., 1995. Clustering full text documents. *Proceedings of the IJCAI Workshop on Data …*, pp.1–10.

McDonald, N., 2010. *Final Periodic Activity Report: HILAS*,

McDonald, N. et al., 2000. Safety management systems and safety culture in aircraft maintenance organisations. In *Safety Science*. pp. 151–176.

Miller, G. a., 1995. WordNet: a lexical database for English. *Communications of the ACM*, 38(11), pp.39–41.

Mitchell, T.M. & others, 1997. Machine learning.

Muller, A., Crespo Marquez, A. & Iung, B., 2008. On the concept of e-maintenance: Review and current research. *Reliability Engineering and System Safety*, 93(8), pp.1165–1187.

Nadkarni, P.M., Ohno-Machado, L. & Chapman, W.W., 2011. Natural language processing: an introduction. *Journal of the American Medical Informatics Association*, 18(5), pp.544–551.

Natarajan, J. et al., 2005. Knowledge discovery in biology and biotechnology texts: a review of

techniques, evaluation strategies, and applications. *Critical reviews in biotechnology*, 25(1–2), pp.31–52.

Perrin, P. & Petry, F.E., 2003. Extraction and representation of contextual information for knowledge discovery in texts. *Information Sciences*, 151(SUPPL), pp.125–152.

Pradhan, S. et al., 2004. Shallow Semantic Parsing Using Support Vector Machines. *Proceedings of the Human Language Technology Conference/North American chapter of the Association of Computational Linguistics*.

Rajpathak, D.G., 2013. An ontology based text mining system for knowledge discovery from the diagnosis data in the automotive domain. *Computers in Industry*, 64(5), pp.565–580.

Ramakrishnan, C. et al., 2012. Layout-aware text extraction from full-text PDF of scientific articles. *Source Code for Biology and Medicine*, 7(1), p.7.

van Rijsbergen, C.J., 1986. A non-classical logic for information retrieval. *The Computer Journal*, 29(6), pp.481–485.

Russell, S.J. et al., 1995. *A Modern Approach*,

Salton, G., 1970. Evaluation problems in interactive information retrieval. *Information Storage and Retrieval*, 6(1), pp.29–44.

Salton, G. & Buckley, C., 1988. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5), pp.513–523.

Schank, R. & Abelson, R.P., 1977. Scripts, Plans, Goals and Understanding: An Introduction into Human Knowledge Structures.

Sebastiani, F., 2002. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1), pp.1–47.

Small, S. & Medsker, L., 2014. Review of information extraction technologies and applications. *Neural Computing & Applications*, 25(3/4), pp.533–548.

Sokolova, M. & Lapalme, G., 2009. A systematic analysis of performance measures for classification tasks. *Information Processing and Management*, 45(4), pp.427–437. Available at: http://dx.doi.org/10.1016/j.ipm.2009.03.002.

Spasic, I. et al., 2014. Text mining of cancer-related information: Review of current status and future directions. *International Journal of Medical Informatics*, 83(9), pp.605–623.

Stapley, B.J., Kelley, L.A. & Sternberg, M.J., 2001. Predicting the sub-cellular location of proteins from text using support vector machines. In *Proceedings of the 7th Pacific Symposium on Biocomputing*. pp. 374–385.

Taylor, M., 2008. *Final Report - TATEM (Technologies and Techniques for New Maintenance Concepts). FP6 – Aerospace, Project reference: 502909, Brussels.*,

Thilmany, J., 2004. Information in order. *Mechanical Engineering*, 126(9), p.46.

Toutanova, K. et al., 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - NAACL '03*. Morristown, NJ, USA: Association for Computational Linguistics, pp. 173–180.

Tretten, P. & Karim, R., 2014. Enhancing the usability of maintenance data management systems. *Journal of Quality in Maintenance Engineering*, 20(3), pp.290–303.

Uğuz, H., 2011. A two-stage feature selection method for text categorization by using information gain, principal component analysis and genetic algorithm. *Knowledge-Based Systems*, 24(7), pp.1024–1032.

Verhagen, W.J.C. & Curran, R., 2013. An ontology-based approach for aircraft maintenance task support. *20th ISPE International Conference on Concurrent Engineering, CE 2013 - Proceedings*, pp.494–506.

Voorhees, E.M., 2007. TREC: Continuing information retrieval's tradition of experimentation. *Communications of the ACM*, 50(11), pp.51–54.

Wimalasuriya, D. & Dou, D., 2010. Ontology-based information extraction: An introduction and a survey of current approaches. *Journal of Information Science*, 36(3), pp.306–323.

Wylie, R. et al., 1997. IDS: improving aircraft fleet maintenance. *Innovative Applications of Artificial Intelligence - Conference Proceedings*, 14(3), pp.1078–1085.

Yang, Y. & Pedersen, J.O., 1997. A Comparative Study on Feature Selection in Text Categorization. *Proceedings of the 14th International Conference on Machine Learning*, 20(15), pp.412–420.

Zafiharimalala, H., Robin, D. & Tricot, A., 2014. Why Aircraft Maintenance Technicians Sometimes Do Not Use Their Maintenance Documents: Towards a New Qualitative Perspective. *The International Journal of Aviation Psychology*, 24(3), pp.190–209.

Appendix 1: Ontology Interface Script

```
//Initial loads
var fs = require('fs');

//Load all files in the ontology directory
var onto_directory = './ontology/';

//Create variables
var ontodata_raw = '';
var ontodata = {};
ontodata.info = '';
ontodata.declarations = {};
ontodata.declarations.classes = [];
ontodata.declarations.properties = [];
ontodata.equiclasses = {};
ontodata.subclasses = [];
ontodata.annotations = [];
ontodata.sampledocs = [];

//All ontology scripts need to be created in the modules.export section
module.exports = {

  load: function(numberdocs,callback) {
    //Load all files in the ontology directory
    //var tempdata = '';
    var onto_files = fs.readdirSync(onto_directory);

    //Get filename of the last file in the directory
    var onto_file_newest = onto_files[onto_files.length-1];

    fs.readFile(onto_directory+onto_file_newest, 'utf8', function (err, data) {
      if (err) {
          return console.log(err);
      } else {
          console.log('>> Ontology Scripts: Creating ontology structure');
          ontodata_raw = data;

          console.log('>> Ontology Scripts: Loading ontology data');

          //1) Obtain the standard info string
          var owl_declaration_first = ontodata_raw.indexOf('<Declaration>');
          ontodata.info = ontodata_raw.slice(0,owl_declaration_first-2);

          //2) Obtain subclasses
          var owl_equiclasses_first = ontodata_raw.indexOf('<EquivalentClasses>');
          var ontodata_declarations = ontodata_raw.slice(owl_declaration_first,owl_equiclasses_first);
          var owl_subclass_first = ontodata_raw.indexOf('<SubClassOf>');
          var ontodata_equiclasses = ontodata_raw.slice(owl_equiclasses_first,owl_subclass_first);
          var owl_subobjproperty_first = ontodata_raw.indexOf('<SubObjectPropertyOf>');
          var ontodata_subclasses = ontodata_raw.slice(owl_subclass_first,owl_subobjproperty_first);

          var k = 0;
          var subclass_loadlimit_counter = 0;
          var subclasses_yesorno = {};

          while((k = ontodata_subclasses.indexOf('<SubClassOf>', k)) > -1) {
              //Cut off subclass piece
              var subclass_start = ontodata_subclasses.indexOf('<SubClassOf>', k) + 12;
              var subclass_end = ontodata_subclasses.indexOf('</SubClassOf>', k);
              var subclass_data = ontodata_subclasses.slice(subclass_start,subclass_end);

              //Define positions and cut off points
              var subclass_cutoff_subclass_1 = subclass_data.indexOf('"')+1;
              var subclass_cutoff_subclass_2 = subclass_data.indexOf('"',subclass_cutoff_subclass_1);
              var subclass_subclass = subclass_data.slice(subclass_cutoff_subclass_1,subclass_cutoff_subclass_2);

              var subclass_cutoff_topclass_1 = subclass_data.indexOf('"', subclass_cutoff_subclass_2+1)+1;
              var subclass_cutoff_topclass_2 = subclass_data.indexOf('"',subclass_cutoff_topclass_1);
              var subclass_topclass = subclass_data.slice(subclass_cutoff_topclass_1,subclass_cutoff_topclass_2);

              if(subclass_topclass == '#DocumentID') {
                  if(subclass_loadlimit_counter <= numberdocs || numberdocs == 0) {
                      ontodata.subclasses.push({"subclass": subclass_subclass, "topclass": subclass_topclass});
                      subclasses_yesorno[subclass_subclass] = true;
                      subclass_loadlimit_counter++;
                  } else {
                      subclasses_yesorno[subclass_subclass] = false;
                  }

              } else {
                  ontodata.subclasses.push({"subclass": subclass_subclass, "topclass": subclass_topclass});
```

```
                subclasses_yesorno[subclass_subclass] = true;
            }

            k++;
        }

        //3) Obtain the classes and properties
        var i = 0;
        while((i = ontodata_declarations.indexOf('<Declaration>', i)) > -1) {
            //Cut off declaration piece
            var declaration_start = ontodata_declarations.indexOf('<Declaration>', i);
            var declaration_end = ontodata_declarations.indexOf('</Declaration>', i);
            var declaration_data = ontodata_declarations.slice(declaration_start,declaration_end);

            //Define positions and cut off points
            var class_position = declaration_data.indexOf('Class');
            var property_position = declaration_data.indexOf('ObjectProperty');
            var cut_off_point_start = declaration_data.indexOf('"')+1;
            var cut_off_point_end = declaration_data.lastIndexOf('"');
            var declaration_correct = declaration_data.slice(cut_off_point_start,cut_off_point_end)

            //Store declaration data in ontodata object
            if(class_position > -1) {
                if(subclasses_yesorno[declaration_correct]) {
                    ontodata.declarations.classes.push(declaration_correct);
                }
            } else if(property_position > -1) {
                ontodata.declarations.properties.push(declaration_correct);
            }

            i++;
        }

        //4) Obtain equivalent classes
        var j = 0;
        while((j = ontodata_equiclasses.indexOf('<EquivalentClasses>', j)) > -1) {
            //Cut off equiclass piece
            var equiclasses_start = ontodata_equiclasses.indexOf('<EquivalentClasses>', j) + 19;
            var equiclasses_end = ontodata_equiclasses.indexOf('</EquivalentClasses>', j);
            var equiclasses_data = ontodata_equiclasses.slice(equiclasses_start,equiclasses_end);

            //Define positions and cut off points
            var equiclass_cutoff_class_1 = equiclasses_data.indexOf('"')+1;
            var equiclass_cutoff_class_2 = equiclasses_data.indexOf('"',equiclass_cutoff_class_1);
            var equiclass_class = equiclasses_data.slice(equiclass_cutoff_class_1,equiclass_cutoff_class_2);

            var equiclasses_type_some = equiclasses_data.indexOf('ObjectSomeValuesFrom');
            var equiclasses_type_all = equiclasses_data.indexOf('ObjectAllValuesFrom');
            var equiclass_type = "";
            if(equiclasses_type_all > -1) {
                equiclass_type = "All";
            } else {
                equiclass_type = "Some";
            }

            var equiclass_cutoff_property_1 = equiclasses_data.indexOf('"',equiclass_cutoff_class_2+1)+1;
            var equiclass_cutoff_property_2 = equiclasses_data.indexOf('"',equiclass_cutoff_property_1);
            var equiclass_property =
            equiclasses_data.slice(equiclass_cutoff_property_1,equiclass_cutoff_property_2);

            var equiclass_cutoff_propertyclass_1 =
            equiclasses_data.indexOf('"',equiclass_cutoff_property_2+1)+1;
            var equiclass_cutoff_propertyclass_2 =
            equiclasses_data.indexOf('"',equiclass_cutoff_propertyclass_1);
            var equiclass_propertyclass =
            equiclasses_data.slice(equiclass_cutoff_propertyclass_1,equiclass_cutoff_propertyclass_2);

            if(subclasses_yesorno[equiclass_class]) {                        //Only include all 'wanted'
            subclasses
          if(ontodata.equiclasses.hasOwnProperty(equiclass_class) == false) {
            ontodata.equiclasses[equiclass_class] = [];
          }

         ontodata.equiclasses[equiclass_class].push({"type": equiclass_type, "property": equiclass_property,
         "propertyclass": equiclass_propertyclass});
            }
            j++;
        }

        //5) Obtain annotations
```

```
        var owl_annotation_first = ontodata_raw.indexOf('<AnnotationAssertion>');
        var owl_annotation_last = ontodata_raw.indexOf('</Ontology>');
        var ontodata_annotations = ontodata_raw.slice(owl_annotation_first,owl_annotation_last);

        var a = 0;
        while((a = ontodata_annotations.indexOf('<AnnotationAssertion>', a)) > -1) {
            //Cut off annotation piece
            var annotation_start = ontodata_annotations.indexOf('<AnnotationAssertion>', a) + 21;
            var annotation_end = ontodata_annotations.indexOf('</AnnotationAssertion>', a);
            var annotation_data = ontodata_annotations.slice(annotation_start,annotation_end);

            //Define positions and cut off points
            var annotation_cutoff_iri_1 = annotation_data.indexOf('<IRI>')+5;
            var annotation_cutoff_iri_2 = annotation_data.indexOf('</IRI>');
            var annotation_iri = annotation_data.slice(annotation_cutoff_iri_1, annotation_cutoff_iri_2);

            var annotation_cutoff_content_1 = annotation_data.indexOf('XMLSchema#string">')+18;
            var annotation_cutoff_content_2 = annotation_data.indexOf('</Literal>');
            var annotation_content =
            annotation_data.slice(annotation_cutoff_content_1,annotation_cutoff_content_2);

            ontodata.annotations.push({"doc_class": annotation_iri, "content": annotation_content});

            a++;
        }
    }
    callback(ontodata);
  });
},

log: function() {
  console.log('>> Ontology Scripts: Logging ontology data');
  console.log(ontodata);
},

save: function(new_ontodata, callback) {
  console.log('>> Ontology Scripts: Saving ontology data');

  //Save new ontodata in current ontodata file
  ontodata = new_ontodata;

  //Create OWL file from Ontodata
  var ontodata_text = '';

  //Add standard info string
  ontodata_text += ontodata.info;

  //Add declarations: classes
  console.log('>> Ontology Scripts: Saving classes');
  for(var i = 0; i < ontodata.declarations.classes.length; i++) {
    ontodata_text += '\n\t<Declaration>';
    ontodata_text += '\n\t\t<Class IRI="'+ontodata.declarations.classes[i]+'"/>';
    ontodata_text += '\n\t</Declaration>';
  }

  //Add declarations: properties
  console.log('>> Ontology Scripts: Saving properties');
  for(var j = 0; j < ontodata.declarations.properties.length; j++) {
    ontodata_text += '\n\t<Declaration>';
    ontodata_text += '\n\t\t<ObjectProperty IRI="'+ontodata.declarations.properties[j]+'"/>';
    ontodata_text += '\n\t</Declaration>';
  }

  //Add equivalent classes
  console.log('>> Ontology Scripts: Saving equivalent classes');
  for(var k = 0; k < Object.keys(ontodata.equiclasses).length; k++) {
    var equiclass_topclass = Object.keys(ontodata.equiclasses)[k];

    for(var l = 0; l < ontodata.equiclasses[equiclass_topclass].length; l++) {
      var equiclass_info = ontodata.equiclasses[equiclass_topclass][l];
      ontodata_text += '\n\t<EquivalentClasses>';
      ontodata_text += '\n\t\t<Class IRI="'+equiclass_topclass+'"/>';
      ontodata_text += '\n\t\t<Object'+equiclass_info.type+'ValuesFrom>';
      ontodata_text += '\n\t\t\t<ObjectProperty IRI="'+equiclass_info.property+'"/>';
      ontodata_text += '\n\t\t\t<Class IRI="'+equiclass_info.propertyclass+'"/>';
      ontodata_text += '\n\t\t</Object'+equiclass_info.type+'ValuesFrom>';
      ontodata_text += '\n\t</EquivalentClasses>';
    }
  }
```

```
234     //Add subclasses
235     console.log('>> Ontology Scripts: Saving subclasses');
236     for(var m = 0; m < ontodata.subclasses.length; m++) {
237       ontodata_text += '\n\t<SubClassOf>';
238       ontodata_text += '\n\t\t<Class IRI="'+ontodata.subclasses[m].subclass+'"/>';
239       ontodata_text += '\n\t\t<Class IRI="'+ontodata.subclasses[m].topclass+'"/>';
240       ontodata_text += '\n\t</SubClassOf>';
241     }
242
243     //Add subject object properties
244     for(var n = 0; n < ontodata.declarations.properties.length; n++) {
245       ontodata_text += '\n\t<SubObjectPropertyOf>';
246       ontodata_text += '\n\t\t<ObjectProperty IRI="'+ontodata.declarations.properties[n]+'"/>';
247       ontodata_text += '\n\t\t<ObjectProperty abbreviatedIRI="owl:topObjectProperty"/>';
248       ontodata_text += '\n\t</SubObjectPropertyOf>';
249     }
250
251     //Add functional properties
252     for(var o = 0; o < ontodata.declarations.properties.length; o++) {
253       ontodata_text += '\n\t<FunctionalObjectProperty>';
254       ontodata_text += '\n\t\t<ObjectProperty IRI="'+ontodata.declarations.properties[o]+'"/>';
255       ontodata_text += '\n\t</FunctionalObjectProperty>';
256     }
257
258     //Add annotations
259     for(var a = 0; a < ontodata.annotations.length; a++) {
260       ontodata_text += '\n\t<AnnotationAssertion>';
261       ontodata_text += '\n\t\t<AnnotationProperty abbreviatedIRI="rdfs:comment"/>';
262       ontodata_text += '\n\t\t<IRI>'+ontodata.annotations[a].doc_class+'</IRI>';
263       ontodata_text += '\n\t\t<Literal
          datatypeIRI="http://www.w3.org/2001/XMLSchema#string">'+ontodata.annotations[a].content+'</Literal>';
264       ontodata_text += '\n\t</AnnotationAssertion>';
265     }
266
267     //Add concluding standard info
268     ontodata_text += '\n</Ontology>\n\n\n\n<!-- Generated by the OWL API (version 4.2.1.20160306-0033)
       https://github.com/owlcs/owlapi -->';
269
270     //Load all files in the ontology directory
271     var onto_files = fs.readdirSync(onto_directory);
272
273     //Get filename of the last file in the directory
274     var onto_file_newest = onto_files[onto_files.length-1];
275
276     //Get current date
277     var current_date    = new Date();
278     var current_date_year   = current_date.getFullYear();
279     var current_date_month  = current_date.getMonth()+1;
280     if(current_date_month < 10) { var current_date_month_str = '0'+current_date_month; } else { var
       current_date_month_str = current_date_month; }
281     var current_date_day  = current_date.getDate();
282     if(current_date_day < 10) { var current_date_day_str = '0'+current_date_day; } else { var current_date_day_str
       = current_date_day; }
283
284     //Add date to the new filename
285     var filename_new    = 'MacsterOntology_';
286     filename_new      += current_date_year + current_date_month_str + current_date_day_str + '_v';
287
288     //Extract components of file name
289     var onto_file_year    = parseInt(onto_file_newest.substr(16,4));
290     var onto_file_month   = parseInt(onto_file_newest.substr(20,2));
291     var onto_file_month_str = onto_file_newest.substr(20,2);
292     var onto_file_day     = parseInt(onto_file_newest.substr(22,2));
293     var onto_file_day_str = onto_file_newest.substr(22,2);
294     var onto_file_version   = parseInt(onto_file_newest.substr(26,(onto_file_newest.indexOf('.')-26)));
295
296     //Add the new version number to the filename
297     if(onto_file_year < current_date_year || onto_file_month < current_date_month || onto_file_day <
       current_date_day) {
298       filename_new += '001';
299     } else {
300       if(onto_file_version < 9) {
301         filename_new += '0';
302       }
303       if(onto_file_version < 99) {
304         filename_new += '0';
305       }
306       filename_new += String(onto_file_version+1);
307     }
308
```

```
    console.log('>> Ontology Scripts: Saving the owl file');
    fs.writeFile(onto_directory+filename_new+'.owl', ontodata_text, function(err){
      if(err) throw err;
      callback();
    });
  },

  search: function (search_options,callback) {
    //Check if registration mark exists
    var data_lookedup = JSON.parse(search_options);

    var regis_lookup = '#'+data_lookedup["Registration Mark"];
    var data_returned = '';

    data_lookedup["Aircraft Manufacturer"] = '';
    data_lookedup["Aircraft Type"] = '';

    if(ontodata.equiclasses[regis_lookup] !== undefined) {
      data_returned = {};

      for(var l = 0; l < ontodata.equiclasses[regis_lookup].length; l++) {
        var equiclass_info = ontodata.equiclasses[regis_lookup][l];
        if(equiclass_info.property == '#hasAircraftManufacturer') {
          data_lookedup["Aircraft Manufacturer"] = equiclass_info.propertyclass;
        }
        if(equiclass_info.property == '#hasAircraftType') {
          data_lookedup["Aircraft Type"] = equiclass_info.propertyclass;
        }
      }
    }

    var doc_data = [];

    for(var i = 0; i < sample_docs.length; i++) {
      var sample_doc = sample_docs[i];
      var includecheck_AcType = false;
      var includecheck_ATA  = false;
      for(var l = 0; l < ontodata.equiclasses[sample_doc].length; l++) {
        var equiclass_info = ontodata.equiclasses[sample_doc][l];
        //Select only searchable (so with a/c type of search)
        if(equiclass_info.property == '#hasDocumentAircraftType') {
          if(equiclass_info.propertyclass == data_lookedup["Aircraft Type"]) {
            includecheck_AcType = true;
          }
        }
        if(equiclass_info.property == '#hasDocumentATAChapter') {
          var ATAChapter = '';
          if(data_lookedup["Topic"] == "Fuel (ATA 28)") {
            ATAChapter = '#Number28';
          } else if(data_lookedup["Topic"] == "Fuselage (ATA 53)") {
            ATAChapter = '#Number53';
          } else if(data_lookedup["Topic"] == "Wings (ATA 57)") {
            ATAChapter = '#Number57';
          }
          if(equiclass_info.propertyclass == ATAChapter) {
            includecheck_ATA = true;
          }
        }
      }
      if(includecheck_AcType && includecheck_ATA) {
        var docname = sample_doc.substr(9,(sample_doc.length-15));

        var docname_read = sample_doc.substr(1,(sample_doc.length-5));
        ontodata.equiclasses[sample_doc].push({"type": "All", "property": "#hasDocumentName", "propertyclass":
        docname});
        ontodata.equiclasses[sample_doc].push({"type": "All", "property": "#hasDocumentNameFull", "propertyclass":
        docname_read});

        doc_data.push(ontodata.equiclasses[sample_doc]);

      }
    }

    data_lookedup["Documents"] = doc_data;

    callback(data_lookedup);
  }
};
```