UNIVERSITY OF TURKU

Extractive Summarization

Experimental work on nursing notes in Finnish

Health Technology

Master's Degree Programme in Digital Health and Life Sciences

Department of Computing, Faculty of Technology

Master of Science Thesis

Author:

Farzana Tajuddin

Supervisors:

Dr. Antti Airola

Dr. Hans Moen

Dr. Laura -Maria Peltonen

March 2022

UNIVERSITY OF TURKU

Master of Science Thesis
Department of Computing, Faculty of Technology
University of Turku

Subject: Health Technology
Programme: Master's Degree Programme in Digital Health and Life Sciences
Author: Farzana Tajuddin
Title: Extractive Summarization: Experimental work on nursing notes in Finnish
Number of pages: 51 pages, 14 appendix pages
Date: March 2022

Natural Language Processing (NLP) is a subfield of artificial intelligence and linguistics that is concerned with how a computer machine interacts with human language. With the increasing computational power and the advancement in technologies, researchers have been successful at proposing various NLP tasks that have already been implemented as real-world applications today. Automated text summarization is one of the many tasks that has not yet completely matured particularly in health sector. A success in this task would enable healthcare professionals to grasp patient's history in a minimal time resulting in faster decisions required for better care.

Automatic text summarization is a process that helps shortening a large text without sacrificing important information. This could be achieved by paraphrasing the content known as the abstractive method or by concatenating relevant extracted sentences namely the extractive method. In general, this process requires the conversion of text into numerical form and then a method is executed to identify and extract relevant text.

This thesis is an attempt of exploring NLP techniques used in extractive text summarization particularly in health domain. The work includes a comparison of basic summarizing models implemented on a corpus of patient notes written by nurses in Finnish language. Concepts and research studies required to understand the implementation have been documented along with the description of the code.

A python-based project is structured to build a corpus and execute multiple summarizing models. For this thesis, we observe the performance of two textual embeddings namely Term Frequency - Inverse Document Frequency (TF-IDF) which is based on simple statistical measure and Word2Vec which is based on neural networks. For both models, LexRank, an unsupervised stochastic graph-based sentence scoring algorithm, is used for sentence extraction and a random selection method is used as a baseline method for evaluation.

To evaluate and compare the performance of models, summaries of 15 patient care episodes of each model were provided to two human beings for manual evaluations. According to the results of the small sample dataset, we observe that both evaluators seem to agree with each other in preferring summaries produced by Word2Vec LexRank over the summaries generated by TF-IDF LexRank. Both models have also been observed, by both evaluators, to perform better than the baseline model of random selection.

Key words: Natural Language Processing, Text Summarization, Nursing Notes, Sentence Level Extraction, Extractive Summarization, Finnish

Acknowledgements

On completion of this work, I would first like to thank the Most Gracious and the Most Merciful, Allah Almighty, for giving me the strength and ability. I would then like to express my sincere gratitude to my supervisors Antti Airola, Hans Moen and Laura-Maria Peltonen for giving me the guidance, constructive feedbacks and encouragement along with the opportunity of exploring this topic. I would also like to thank the two evaluators* who volunteered their participation in reviewing the summaries generated by the system and providing their invaluable feedbacks.

I am greatly thankful of my manager, Kirill Inkine, my colleagues and my friends at my workplace for their understanding, patience and support during this thesis. I would also like to thank people working in Turku cafés for their hospitality and their special arrangements for students. I am, of course, grateful of my close friends and family for their abundant support and encouragement through-out this work. Special thanks to my dearest parents without whom this would not have been possible.

* The names are undisclosed to preserve confidentiality

Table of contents

List of Tables

List of Figures

1 Introduction

Healthcare professionals such as nurses are known to have a difficult time using the electronic health records whilst taking care of patients [1]. From the many time-consuming tasks nurses have, one of them is writing patient discharge summaries of nursing notes taken during patient's stay at a hospital. Providing the most important content from multiple notes in a summary form could help in accomplishing such tasks and thus reducing a substantial time and effort for healthcare professionals.

Natural Language Processing is a field that studies the interaction of computer machines with human language. It borrows the concepts from Linguistics and uses statistical, machine learning and deep learning tools to solve problems such as producing concise text - Automatic Text Summarization. The problem falls into two major categories: identifying a smaller set of important sentences known as extractive summarization or generating new concise text known as abstractive summarization.

MITRE Text and Audio Processing (MiTAP) is an example of an early system that uses named entity recognition, machine translation and machine learning techniques to extract important sentences from single-document, multi-document, multi-lingual and multimedia.[2] CliniText system is an example of a recent system that takes input various clinical raw data and generates a summary using abstractive technique [3].

The objective of this work is part of a greater goal of building and implementing a text summarization application that could ultimately assist health care professionals in their daily work. As an initial phase, background study about the topic and related work is explored. A python-based project is built to implement some known extractive summarization techniques on patient's nursing notes.

Given that there are so many techniques and a wide range of parameter values that could be selected, the number of combinations for the possible implementation are immense. For this thesis, we will explore the comparison between word2vec and term frequency – inverse document frequency (TF-IDF) feature selection using Lex Ranking algorithm for sentence extraction. Random selection of sentences will be used as a baseline model.

Word2vec and TF-IDF are machine learning algorithms used in learning features of a sentence. The features are then used to rank the importance of a sentence. Ideally, they both should perform better than the baseline method of just randomly selecting sentences.

Word2vec learns the representation of words looking at its surrounding words, whilst TF-IDF learns the frequency based features looking at the whole corpus. We believe Word2vec model capturing word semantics, should perform better than TF-IDF when keeping all other environmental variables similar.

Thus, we presume the following two statements which we can affirm after analysing our results:

1. Word2vec_LexRank and TF-IDF_LexRank methods will have better scores than our baseline model of random selection.
2. Word2vec_LexRank will give better results as compared to TF-IDF_LexRank.

The intention of this document is to:

- address the background information required to understand the work done for this thesis (Chapter 2)
- give an overview of the related work done by other researchers. (Chapter 3)
- elaborate on the practical work including the data, experimental methods and discuss the outcomes (Chapter 4)
- discuss the limitations, future work and conclude the thesis (Chapter 5)
- document the code (Appendix)

2 Background

2.1 Natural Language Processing

Human brain receives signals from the five basic senses to help us understand and perceive the world around us [4]. Artificial Intelligence is a field of study that is concerned with how human abilities can be incorporated into machines. Machine learning algorithms including deep learning are techniques that enable machines the ability to learn. Linguistics is the scientific study of language that includes the study of phonetics, phonology, morphology, syntax, semantics and the pragmatics.

Natural Language Processing (NLP) is a sub field of Artificial Intelligence and Linguistics that uses computational techniques to enable computer machines the ability to process human languages which includes reading, writing, speaking and responding appropriately which involves understanding and perceiving like humans.

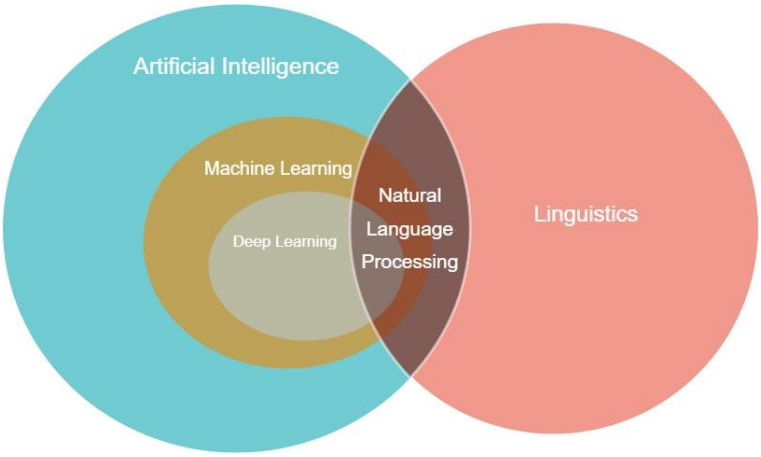

Figure 1: Representing Natural Language Processing as a subset of Linguistics and Artificial Intelligence

Computational techniques used for NLP initially involved using explicit algorithms based on counts such as Term Frequency – Inverse Document Frequency. With the advancement in machine learning algorithms, supervised and unsupervised algorithms such as support vector machines and neural networks (Word2Vec) were implemented for both classification and clustering problems. With the increased complexity in layering the neural networks such as recurrent neural networks and the recent development of language models like BERT, as part of deep learning, NLP techniques seem to have now evolved to give results closer in depicting a human brain.

2.1.1 The Five Phases

NLP generally comprises of five major phases which include lexical analysis, syntactic analysis, semantic analysis, discourse integration and pragmatic analysis.

Lexical analysis or morphological analysis refers to the analysis of individual words or tokens. This includes splitting text into its smallest unit (words or tokens), the identification of common words also known as stop words, the realization of the different forms and descriptions of the words. The common NLP tasks used in this phase are known as tokenization, stop word removal and lemmatization.

Syntactical analysis or parsing is concerned with the study of how the words can be placed together to form a structured sentence and how one word is related to other words in the sentence. This means ensuring the basic grammatical rules of a language such as the ordering of noun, verb etc. This requires analysing several sentences of the same word. The corresponding NLP task is also referred as parts of speech tagging.

Semantic analysis is concerned with the meaning of the words based on the context used in the sentence. The resultant of this analysis is a meaningful sentence. Sentences that are syntactically or grammatically correct may not necessarily have a meaningful sense. The building blocks for this analysis are the classification of entities, concepts, relations and predicates.

Discourse Analysis is the study of a sentence with respect to other preceding or following sentences. It enables the meaning of the sentence in relation to the context of the paragraph/ document. It is important to ensure that when an NLP system generates a paragraph, the sentences follow a specific discourse to give a meaningful understanding of the context as a whole.

Pragmatic Analysis is concerned with understanding a sentence or set of sentences with respect to prior knowledge from external documents. This involves interpreting the meaning of the context with respect to the different situations that may be determined by the varying location, time or social content.

2.1.2 Tasks and Applications

Today, NLP techniques have made the following tasks possible. Some of these tasks have reached the maturity level and are already being used in our daily lives whilst others require more work.

Machine Translation: the ability to convert speech and text from one human language to another. This task involves understanding the text and then generating new meaningful text in the correct structure as per the rules of the new language.

Semantic Role Labelling: enables forming the meaning of words in relation to other words. It enables creating a relationship structure between words illustrating the predicate- argument structures.

Text/Document Classification: helps in sorting similar documents or similar topic related sentences into the same category. This can be used for spam detection.

Information Retrieval: Retrieving intended information from unstructured text using semantic search, recall retching, question and answering techniques of NLP. Sentimental Analysis could be used to retrieve an analysis of the social media content.

Natural Language Generation: involves the creation of speech and text to form meaning full sentences and paragraphs. This task requires the language to be semantically and syntactically correct containing the right discourse of the context and in certain situations the right pragmatics as well.

Text Summarization: enables creating a concise meaningful and important text extracted or generated from a single large document of text or multiple documents.

Relationship Extraction: involves creating semantic relationships between two or more entities present in a sentence. This includes the process of recognizing the different entities and phrases and annotating text into different parts of speech.

Topic Modelling: is the process of identifying the different topics present in a corpus. This requires the NLP model to have the capability of reading comprehension and classification. It is text-mining statistical modelling used to discover hidden semantic structures within a document.

Speech Recognition: the ability to recognize and process human spoken language. The spoken words can be converted to written text. Examples include Alexa, Siri, Bixby, etc. that understand respond back to many human queries.

2.1.3 Challenges

Human language is quite diverse and complex in nature. There are 35 different languages used on the internet today [5] with their own phonemes, morphemes, lexemes, syntactical structure and semantics.

NLP processes made for one language would require customizations to be used for another language since every language has its own complexity. Languages such as English and Finnish have spaces to segregate words, whereas it is difficult to identify delimiters in languages such as Chinese. English, French, Chinese are languages written from left to right whereas Arabic, Urdu and Persian are read from right to left. NLP processes made for multilingual inputs may have additional challenges. A word in one language may have a different meaning in another language. For example, “bandar” in Arabic means beautiful whereas refers to a monkey in Urdu.

Building an NLP process for a single language is by itself complicated. Apart from the complexities that may be unique to each language, every language has a set of dictionary words that keeps increasing and changing. Words may have different meanings based on different contexts for example in English, the word “tablet” could refer to the medicine or the electronic device commonly confused even by humans. Identifying idioms, slang words, phrases, sarcasm or understanding abbreviations such as NLP for “natural language processing” or “neuro-linguistic programming” are complicated as there are no defined rules. Spoken languages have their own challenges as there are different dialects for the same language.

Developing a generalized NLP process for every domain is also a challenge today. There are differences in vocabulary and writing or speaking styles in different domains. For example, in healthcare domain, when working on patient data, the processes may differ as professionals have their own medical terminologies, phrases and abbreviations used in their daily communication. Formal documents such as news or academic writing may have their own pre-processing requirements.

2.2 Text Summarization

Text Summarization, also known as Automatic Summarization, is one of the NLP tasks that aims at replacing existing text with a concise and comprehensive version of the actual text. The idea first began in 1958 by Hans Peter Luhn who introduced a statistical based extractive summarization method to produce a compact version of technical articles.[6] As the research world shifted from statistical analysis to machine learning, better results were seen.

Below are some of the examples of online summarization applications already available to be used for English language:

- Text Summarizer http://textsummarization.net/text-summarizer
- Quillbot https://quillbot.com/summarize
- Text Compactor https://www.textcompactor.com/
- Resoomer https://resoomer.com/en/

Like any other process, summarizing text also involves an input, processing method and an output. We will discuss the input types as a single document or a multiple document, the processing method as abstractive or extractive, supervised or unsupervised and output as indicative or informative. For our work in this thesis, we are aiming at summarizing multiple nursing documents written in Finnish language using unsupervised extractive methods to obtain informative abstracts.

2.2.1 Single vs Multiple Document

Just like before treating a patient, diagnosis of the problem is important, when summarizing a text, it is important to know the type of input. They may have challenges of their own resulting in different type of solutions.

Multiple documents tend to generally have more repetitive information as compared to a single document. Converting multiple documents may result in losing some important information for example if the order of the documents is important or the time and date.

2.2.2 Indicative vs Informative Abstracts

Text Summarization either helps reader gain a brief understanding of the whole context or with just a few sentences convinces the reader in reading the whole text. The first type of summarization is also called indicative abstracts and the latter is known as informative abstracts. Informative abstracts would be useful for example when working with health notes where the objective is to gain an idea of the whole text. An example for the indicative type of summarization might be an abstract of a content such as news briefs.

Indicative Abstracts: abstracts that allow a searcher to screen the body of literature to decide which documents deserve more detailed attention [7].

2.2.3 Supervised or Unsupervised Learning

When algorithms built to learn parameters are given the expected output, they are known as supervised machine learning algorithms and when they learn without an expected output, they are called unsupervised learning algorithms.
For tasks where the expected output is known, it is preferable to let the machine learn using the results for better performance. However, one must be careful in not overfitting the algorithm or in other words train it using the complete dataset.

2.2.4 Abstractive vs Extractive Summarization

The methods of text summarization fall into two broad categories: Extractive and Abstractive. Extractive method involves isolating few sentences from the actual text usually based on the important set for the sentence. The idea of abstractive method is to focus on the semantics of the context. So paraphrasing or generative new words would be a part of the summarized output of this method.

Following are some of the words and concepts we think one must know to for understanding the work done for this thesis:

2.2.5 Corpus

The complete set of input is called a corpus. This can be a collection of written or spoken texts. In this thesis we refer corpus to a text file containing a collection of all the sentences used for processing the summarization.

2.2.6 Tokens

Any sub-unit of text is called tokens. They could be single words, or a list of words also referred to as sentence tokens.

2.2.7 Tokenization

The process of breaking the text into smaller units. Generally, we have either word tokenizer or sentence tokenizer. Word tokenizer breaks the text into a list of words whiles sentence tokenizer would return a list of sentences. A tokenizer would normally require some instructions on how to perform the splitting of the text. An ideal example for English and Finnish language would be a set of punctuations (full stop, exclamation mark, question mark, etc) for a sentence tokenizer and may be space for a word tokenizer.

2.2.8 Lemmatization

The process of converting words into its original dictionary form is called lemmatization. In languages, we have different forms of words essentially having the same meaning modified only to give a better understanding of the surrounding words in the sentence. It is generally but not always useful to help the computer categorize such words as the same for example: [come, coming, comes, came] or [give, giving, gives, gave]. Sometimes it makes a mistake like converting [cope, coping, copes, coped] to [cope, cop, cope, cop].

2.2.9 Stemming

The process of trimming suffixes or prefixes from a word to bring it closer to the original form is called stemming. Like lemmatization, stemming helps the machine categorize similar words in the same set but instead of changing the form of the word, it just cuts the ends. For example, it would convert [come, coming, comes, came] to [come, come, come, came], [give, giving, gives, gave] to [give, give, give, gave] and [cope, coping, copes, coped] to [cope, cope, cope, cope]. As can be seen, sometimes they are correct and sometimes they are not.

2.2.10 Stop Words

This is a term given to very frequently occurring words that are not important to give any useful information. Removing these words would not make much of a difference to the meaning of the sentence but improve processing time and performance by giving weightage to

more useful words. Examples from the English language would be words like [a, the, but, and, to]. Depending on the task, sometimes these stop words are not removed.

2.2.11 Sentence Vectors

Generally, a human brain reads, translates (if necessary), and comprehends text before summarizing it. In contrast with a human brain, after loading the data into a machine, the text must be transformed into a numeric form such that it can be processed as a basis for forming the summary. Unlike a human brain, a machine requires an algorithm or a learning process.

The transformation of text to numeric form, also commonly known as encoding, word embeddings, vector space modeling or vectorizing the text can be achieved in several ways. The methods would either be statistical based or using machine learning with the output relying on frequencies or prediction-based models. Examples of such methods include Hot-Encoding, Co-occurrence matrix, Term Frequency – Inverse Document Frequency (TF-IDF), Word2vec, Glove. However, since in thesis, we are using TF-IDF and Word2Vec, we will focus on these two in detail.

2.2.11.1 Term Frequency - Inverse Document Frequency

As the name suggests, TF-IDF is a combination of two different methods. Term Frequency represents the number of times the word occurs in the document. Inverse document frequency is the log of the number of documents divided by the number of documents containing the term which basically decreases the weightage of most frequent words.

Example:
Let us say we have two documents each having one sentence:
Document 1 (Doc1) - “This test document test”,
Document 2 (Doc2) - “Document name test”

The term frequencies for each document could be computed as:

Table 1. Example of Term Frequency for Doc1

Tokens	this	test	document
Doc1	1/4	2/4	1/4

Table 2. Example of Term Frequency for Doc2

Tokens	document	name	test
Doc2	$^1/_3$	$^1/_3$	$^1/_3$

The global frequency or IDF for the whole corpus (containing Doc1 and Doc2) could be computed as:

Table 3. Example of Inverse Document Frequency

Terms	**IDF**
Name	Log ($^2/_1$)
This	Log ($^2/_1$)
Test	Log ($^2/_2$)
Document	Log ($^2/_1$)

Combining the Term-Frequency and Inverse Document Frequency, we would get the following document term matrix:

Table 4. Example for TF-IDF

Tokens	this	test	document	name
Doc1	$^1/_4$ * Log ($^2/_1$)	$^2/_4$* Log ($^2/_2$)	$^1/_4$* Log ($^2/_1$)	0
Doc2	0	$^1/_3$* Log ($^2/_2$)	$^1/_3$* Log ($^2/_1$)	$^1/_3$* Log ($^2/_1$)

The documents could be represented as vectors in a high dimensional space where each word is a dimension. Similar documents would have similar words and hence would be closer to each other; the cosine angle of 1 would show the two documents have the same orientation, a cosine angle of 0 would mean there is no similarity.

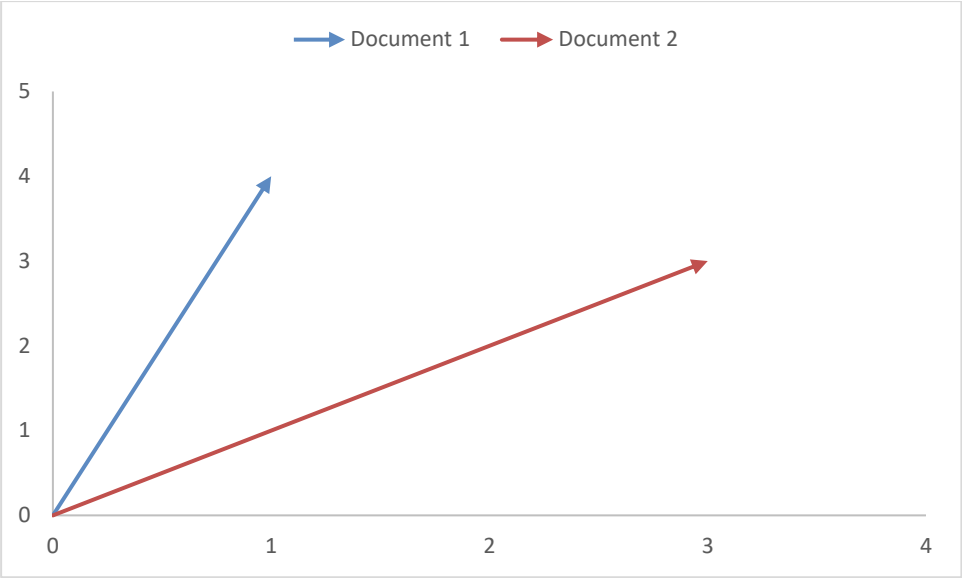

Figure 2: Representing Documents in Vector Space Model

2.2.11.2 Word2vec

Word2vec is a two layered neural network that learns vector representation of words given a corpus of text. It mainly works either by predicting a word given the context as continuous bag of words or by predicting the neighbors of a given word known as the skip-gram model.

Example:
Let us say we have two documents each having one sentence and let's consider the skip-gram model:
Document 1 (Doc1) - "This- essential document test document"
Document 2 (Doc2) – "Document name necessary test"

Considering the window size as 5, we can extract the neigbouring words. The window size of 5 for a particular word means looking at 2 words ahead and 2 words behind. The same can be found below in Table 5. Example for Word2Vec.

Table 5. Example for Word2Vec

Term	Neighbors
This	essential, document
Essential	this, document, test
Document	this, essential, test, document name, necessary
Test	essential, document, name, necessary
Name	document, necessary, test

The input for the neural network would be the combinations of the term with neighbors encoded into 0s and 1s and its weights. The objective is to minimize the loss function such that the prediction output vector contains highest probabilities for the neighboring words. Using backpropagation method, the weights are adjusted resulting in an n- dimensional vector for words which can then be used in building sentence vectors.

2.2.12 Cosine Similarity

When computing similarity between words or sentences represented as vectors, the cosine angle between the vectors is used as a measure to identify if the vectors are pointing to the same direction. To calculate the cosine similarity, we compute the inner product of two vectors normalized to length of 1. [8] This essentially gives a similarity matrix of sentences.

2.2.13 Sentence Rankings

LexRank algorithm gets the concept from PageRank that derives the rank of a sentence based on other sentences. Similarity matrix created by using cosine distance, is used to get the importance a sentence gives to other. Once this score is computed, the top scored sentences can be extracted indicating the most important sentences.

The similarity matrix of sentences can be represented as a graph where the nodes represent the sentences, and the edges represent the value of cosine similarity. In order to find the importance of a sentence, the page rank algorithm can be used.

The formula from [9] for the algorithm can be found below:

$$S(V_i) = (1-d) + d * \sum_{j \in In(V_i)} \frac{1}{|Out(V_j)|} S(V_j)$$

Figure 3. Formula for Ranking Algorithm

When calculating the score for a sentence, weight S(Vi) for a sentence is compared with other sentence S(Vj), incoming and outgoing links are computed. d is the damping factor used for the case if there are no outgoing links. In(Vi) is the set of incoming links and Out(Vj) is the set of outgoing links.

Example:
Let's assume that we have the following three sentences:
Sentence 1 (Sent 1): "This is an important sentence"
Sentence 2 (Sent 2): "This a very important sentence"
Sentence 3 (Sent 3): "When passion turns into work"

Let's further assume that we got the following cosine similarities for the sentence vectors as:

Table 6. Example for Lex Rank

	Sent 1	**Sent 2**	**Sent 3**
Sent 1	1	0.8	0.1
Sent 2	0.8	1	0.2
Sent 3	0.1	0.2	1

Using the formula in Figure 3, and the constant d let's say as 0.85, we can calculate the importance of each sentence through multiple iterations as:

Initial iteration:

Sent1 = (1 - 0.85) + 0.85 * ((0.8/2) + (0.1/2)) = 0.5325

Sent2 = (1 - 0.85) + 0.85 * ((0.8/2) + (0.2/2)) = 0.575

Sent3 = (1 - 0.85) + 0.85 * ((0.1/2) + (0.2/2)) = 0.2775

Next iteration:

Sent1 = (1 - 0.85) + 0.85 * ((0.575/2) + (0.2775/2)) = 0.512

Sent2 = (1 - 0.85) + 0.85 * ((0.5325/2) + (0.2775/2)) = 0.494

Sent3 = (1 - 0.85) + 0.85 * ((0.525/2) + (0.575/2)) = 0.6175

The higher the value of the sentence, the more important the sentence. In our example, the third sentence seems to be the most important sentence. Please note that the values used above are just arbitrary values used for explaining.

2.3 HealthCare

Health Data includes any kind of content related to the medical and health field. This could be a reference content used by healthcare professionals for the treatment of patients or content that is created during the treatment of a patient. The data can be segregated into various categories including audio recordings, images, structured text, unstructured text or even a combination of these.

Electronic Health Records are digital records that provide comprehensive health information about patients. An electronic health record consists of patient's administrative and billing data, patient's demographic information, progress notes, vital signs, medical histories, diagnoses, medications, immunization dates, allergies, radiology images, lab and test results.[10]

Nursing documentation is the formal record detailing the nursing care given to the patient. It is a means of communicating with other healthcare providers, it demonstrates safe and ethical care and for some organizations, meets legal requirements. The documentation mainly consists of nursing history, patient's background information, care plan and drug details using various standard documents such as Nursing Progress Notes, Nursing Admission notes, Nursing Care Plans and Medical Administration Records.

Feblowitz J, Wright A et al. propose a 5 steps model for generating clinical summaries to address many challenges. This includes aggregation (collecting data), organization (structuring the data, sorting the data), reduction and transformation (basically filtering and data cleaning), Interpretation (using domain knowledge) and Synthesis (actions recommendation using clinical standards and guidelines) [11].

Rimma Pivovarov et al discuss challenges in generating clinical summaries using the same framework proposed by Feblowitz et al. [11]. In addition to that they also include various clinical summarization applications discussing their summarization approach, input, output, evaluation methods and additional information [12].

Some of the organizations that help in standardizing nursing documents include Clinical Care Classification (CCC), International Classification of Diseases (ICD), International Classification of Functions (ICF), International Classification for Nursing Practice (ICNP), International Classification of Primary Care (ICPC), Logical Observation Identifiers Names

and Codes (LOINC), North American Nursing Diagnosis Association (NANDA), Nursing Interventions Classification (NIC), Nursing Outcomes Classification (NOC), etc.

Information is recorded from the time the nurse encounters the patient to the time when patient leaves after the completion of the care. Outpatient documentation would be different from inpatients but in both cases, there are some minimal requirements that must be met. In countries like Finland, the legal requirements are enforced. For example, according to the legal regulations in Finland, a patient's treatment in the ward should be recorded daily and there should be a final statement of the treatment containing summaries and follow-up plan.[13]

2.3.1 The Nursing Model

The most common and internationally recognized process used for nursing documentation is based on the nursing model presented by World Health Organization (WHO)[14]. The documents created in the process and the language tools used vary within organizations. The model comprises of the following phases with Diagnosis and Expected Outcome as a separate phase or included in other phases.

1. Assessment

   This is the first phase of the process and is concerned with the initial condition of the patient including subjective and objective information. Subjective information includes elements that cannot be measured but are related to the patient. This includes the information, complaints or expressions the patient shares. Objective information includes quantifiable data such as age, temperature, blood pressure, etc.

2. Diagnosis

   A diagnosis statement, different from medical diagnosis, is built using the information from the assessment phase and clinical judgement of the nurse to reflect the actual or potential condition and needs of the patient. NANDA is an example of a standard language used to document the diagnosis. Some organizations choose this phase to be a part of the previous or latter phase.

3. Planning

   Using the information from the previous phases, a prioritized list is created to ensure life-threatening issues to be resolved before non-life-threatening issues. Measurable and achievable short-term or long-term goals are created by the nurse. These, along with the assessment and diagnosis are written in the patient's record to establish clear communication with other health care professionals.

4. Intervention

   Implementation of the care planned in the previous phase is carried by the nurse, physician or in collaborative with both. The patient's status is re-evaluated and the care plan modified if needed before the treatment. Treatment can include giving medications or performing a procedure. Patient counselling and observing for adverse reactions is part of the process. Each care given is documented in care plan usually with a date-time stamp.

5. Evaluation

   The impact of the nursing interventions is observed and recorded in this phase. The effectiveness of the nursing care plan is critically assessed to see if the care plan and interventions were helpful. Re-assessing the patient's status, if needed, the nursing care plan is modified. Both positive and negative outcomes are recorded.

6. Expected Outcomes

   From the diagnosis, planning and intervention phase, measurable outcomes are created in collaboration with the patient. These outcomes are patient-centered and comprise of characteristic of being specific, measurable, relevant and attainable within specific time frame. Nursing Outcomes Classification (NOC) is an example of a standard language used by nurses to document outcomes to coordinate with the diagnosis established by NANDA.

2.3.2 Ethical Concerns

Ethics is a branch of philosophy that answers questions as to what is morally right or wrong with respect to the decision of behavioural acts to the society.

According to Avasthi A et. al, the four principles of Beauchamp and Childress are fundamental in understanding ethical assessment in health care [15]. The four principles are:

1. Autonomy

   This principle is concerned with giving the independent authority and freedom to the patient to decide on their fate. Patients have the right to accept or refuse a particular treatment and cannot be forced into a particular treatment. To apply this principle, it is particularly important that nursing care plan created for patients and the interventions are in collaboration and consent with the patient.

2. Non-maleficence

   This principle has been noted as the most important principle [16] and is related to preventing any harm to the patient. This includes physical and psychological harm which must be judged by ethical judgement. Informing patients about potential risks and using interventions and re-assessing patient's status timely could prevent harm to patient. For example, resolving adverse effects of a medication.

3. Beneficence

   This principal is based on acts that benefit the patient. Nurses should support patients during their healing and recovery. This includes educating the patient and taking preventive care.

4. Justice

   This principle is based on treating patients fairly, with equality and equity. The treatment given to patients should be irrespective of their background, ethnicity or color, personal character for example.

2.3.3 Challenges

Some of the challenges when it comes to summarizing health text:

1. Sensitivity of Health Care Data

   Since health care data related to patients is confidential and sensitive to be shared publicly, gathering data in this domain is a challenge. Due to this we do not see a lot of research experiments done in this domain. We see a lot of research being done on news briefs and published papers perhaps due to the readily available golden summaries.

2. Complexity of Health Text

   Unlike other standard formal text, healthcare professionals have their own dictionaries. When taking quick notes, they tend to abbreviate words common to them. Abbreviations might also be different for different units within the hospital. There are some important technical words that perhaps should be treated differently, may be given more weightage.

3. Clinical Standard Text

   Health text may already have some numbers that refer to important clinical information for example vital information or ICD codes. When converting text to numbers, we remove the numbers in the preprocessing phase, perhaps we are eliminating some essential information.

4. Impact of missing Information

   Unlike other domains, health domain is crucial in the sense if the summary generated by the system is incorrect or skips some very important information the impact could be life threatening for a patient. Hence it is important to achieve results close to perfection for this domain.

3 Related work

There is more than a decade work done in trying to generate system-based summaries. We have seen the work done in different languages and different domains. For this thesis, we analyse the work done mostly in the last 5 years, but we also list some important methods or work done that has been a great contribution for this topic. For the work done in healthcare domain, we do not filter on the number of years.

Approaches used in summarizing text in a particular domain might not work for another domain. There is a lot of variance in terms of vocabulary and the style of the text structured. In some domains, such as news or research articles, we already find the golden summaries as the news feed or abstract. Whilst in other domains such as healthcare, summaries must be generated or unsupervised approaches must be selected.

Different methods have been applied for summarising texts extracted from various domains including healthcare [17]–[24] patent [25] , research articles [25], emails [26], wikipedia articles[27] and news articles [25], [28]–[39]. Most work on textual summarization has been seen in the news domain.

When summarizing text, words play an important role and with different language, words and their structures change. Different languages may have different best solutions when it comes to summarizing the text. We have found some work done in Turkish [29], Persian [30], Hindi [35], [36], German [27], Finnish [17]–[19] but most work was done in English [21]–[26], [31]–[34], [36], [37], [39]–[41]

In the following sections, we discuss the various methods used for extracting text summarization and the approaches used for evaluating text summarization (Section 3.1 and 3.2 respectively). We then list the various data sets created for the purpose of research in text summarization. (Section 3.3). In the last section we describe the work done in Health Care Domain. (Section 3.4)

3.1 Summarization Methods

There are several methods used in text summarization field however some of the most important or common work done in the history of extractive text summarization are listed below in detail. This includes basic statistical extractive methods such as Luhn's work based on frequency of words, Edmundson's idea of creating frequency dictionaries of local and global words, the work on TF-IDF and then as technology advanced and the use of neural networks became common, Word2Vec or FastText as word embedding model became quite popular in the world of textual summarization. In this list, we also mention the introduction to the graph-based ranking model including LexRank and TextRank.

Statistical based methods were first highlight by Luhn in attempting to summarize technical articles. His work focuses on the importance of the sentence pertaining to the frequency of the words and the relative position of a word in the sentence. He basically suggests that the place where the most frequent words occur close to each other that is where lies the significance of the article. He then clusters sentences that had significant words with not more than 4 non-significant words in between. Significance factor was calculated as the square of the number of significant words divided by the total number of words in the cluster. Most significant sentences were then extracted. His experiment was on 50 articles that had 300 to 4500 words each and 100 people evaluated the work manually. [6]

Local and Global frequency based dictionaries was an idea proposed by Edmundson with his work on summarizing text by enhancing the sentence significance methods. He introduces four new methods namely Cue, Key, Title and Location. He basically uses four different types of dictionaries to obtain sentence scores. Cue and Key dictionaries are extracted from the body where cue contains words from the Corpus and Key only from the current document. Title and Location focus on words from specific location of documents where location is from the set of all documents and Title is from the current document. Sample of 40 documents were manually evaluated for a subjective mean similarity score. A second method of evaluation was using 20 extracts that had 311 sentences to evaluate false negative and false positive scores based on the extraction of worthy sentences. [7]

The idea inverse document frequency was discussed and formulated by Jones. She does so by first discussing the known definitions of exhaustivity and specificity and then aims at redefining them. Exhaustivity, which previously, was the set of various topics of a given document defined by the selected terms, changed to the number of terms a document contains. Specificity, which previously, was the representation of the conceptual or semantic detail of an individual term, changed to the number of documents that contained that term. Her main idea was to introduce a statistical interpretation to specificity. She experiments her idea on three test sets from Aslib Cranfield (200 documents), INSPEC (541 documents), and College of Librarianship Wales projects (797 documents).[42], [43]

Word2Vec was the name of the method used by Tomas Mikolov et al. in their work which involved using a simple two layered neural network to train a large corpus of dataset. They propose two main architectures as CBOW or skip-gram model where the first predicts the current word given the context and the latter predicts the neighbouring words of the given word. They focused on using vectors for words as input that represent the relationship with other words. Google news corpus containing 6B tokens was used for training with a vocabulary size restricted to 1 million most frequent words. Once the model is trained, they show that it produces remarkable results that could be very useful in the natural language processing world. They give an example of an algebraic equation for words which is very commonly read in studies today as King – Man + Woman = Queen. They tested their model both semantically and syntactically [44], [45]

FastText introduced by Piotr Bojanowski et al. is an extension to the skip-gram model work done by Tomas Mikolov et al. They basically propose using the sum of 3-6 n-gram character vector representations of a particular word to represent a word vector. They tested this proposal as two versions; one that considered the vector null for those words that were not in the training set (sisg- Subword Information Skip Gram Negative) and the second as the sum of n-grams for these new words (sisg Subword Information Skip Gram). They experiment word similarity on Arabic, German, English, Spanish, French, Romanian and Russian and word analogy tasks on Czech, German. English and Italian.[46]

Lexical Page Rank or LexRank, a stochastic graph based model for determining the importance of textual units was introduced by Gunes Erkan et al. They propose three different methods for computing centrality in similarity graphs1; degree based, centrality based (considering the importance based on the importance of adjacent nodes), weight based considering the cosine similarities. These methods were used for experimenting on the task of extractive summarization using the common news datasets DUC 2003 and DUC 2004. [47]

TextRank, a graph-based ranking model for selecting important text was introduced by Mihalcea and Tarau in the same year as LexRank was introduced by Gunes. The idea is based on Google's PageRank introduced by Brin and Page in 1999.[48] For NLP, each vertex of the graph would represent the token in consideration. They explain how the score is based on voting or recommendations and experiment the unsupervised model on keyword extraction and sentence extraction. For keyword extraction, they use window sizes of 2,3,5 or 10 words on Inspec database containing 500 abstracts and manually assigned keywords. For sentence extraction, they applied the single document extraction using 567 news articles from the Document Understanding Evaluations 2002 (DUC 2002). [9]

3.2 Evaluation Methods

Evaluating text summarization has seen to be a challenging task mainly because golden summaries (desired output) are not readily available for text other than news or research articles where you already find news briefs and abstracts as the ideal output. In other domains, such as healthcare, either golden summaries are created to enable training models or the summaries need to be manually evaluated. The evaluation results of manual evaluations are expected to be different from person to person as defining a “good summary” is quite subjective which brings in the complexity of measuring the reliability of the results.

Mostly in the presence of golden summaries, Rouge Metrics is used to assess the quality of the summaries along with other basic measures such as precision or sensitivity, F1 scores [18], [19], [21], [22], [25]–[27], [29], [31]–[33], [35], [36], [38], [39], [41]

Rouge (Recall-Oriented Understudy for Gisting Evaluation) is a metric used to evaluate the quality of summaries generated when comparing them with ideal summaries. This method is introduced by Chin-Yew Lin. There are four different types as n-gram cooccurrence statistics (Rouge-N), Longest common subsequence (Rouge-L), Weighted longest common subsequence (Rouge-W) and Skip Bigram Co-occurrence statistics (Rouge-S). To evaluate this measure, the scores are compared to human judgements using the DUC datasets from 2001 to 2003. [49]

To assess the quality of the summaries and predict the extent to which summaries capture the main idea of the text, an automatic scoring model is built by Bortarleanu et al. using machine learning models, language architectures and text complexity indices from the ReaderBench framework. ReaderBench is a software framework designed to enhance the Personal Learning Environment for students and tutors uses textual complexity assessment. [40], [50]

One of the methods used for manually evaluating summaries was capturing scores on a 1 to 4 Likert scale for analytical measures that included assessing whether the summary contained the main idea, the amount of key information found, the structure flow of the summary, appropriate paraphrasing, language beyond the original text, the length of the summary.[40]

Cohen Kappa's score is a statistic usually used to measure the extent to which evaluators are in agreement with each other [23], [32]. Higher scores indicate stronger agreement indicating the ratings given by the evaluators are reliable and can be accepted whilst lower scores require re-assessing. Sometimes, the Inter-rated reliability score is too low to accept the results even after repeated assessments which indicate that the definition of "good quality text" is quite subjective. To resolve this, the test sample is selected based on the most relevant or appropriate case studies ranked by other systems instead of random samples. [23]

3.3 Data Sets

Text Summarization tasks have been performed on various datasets ranging from news, patent documents, healthcare, research articles, emails etc. Datasets in several languages including Turkish, Sindhi, Persian etc are compiled and summaries manually created for the purpose of facilitating research in this area.

Document Understanding Conferences, run by National Institute of Standards and Technology with the main purpose of promoting text summarization tasks have news related collections of data namely from DUC (2001 to 2007). Ranking methods (TextRank and LexRank) combined with four weighted schemes (Jaccard similarity coefficient, TFIDF cosine similarity, Topic signatures similarity, and the Identity similarity measure) were experimented on DUC03 and DUC04 [31]. Meta- Heuristic approach of Shark Smell Optimization was applied on DUC04, DUC06 and DUC07 [35]

Newsroom dataset of 1.3 million summaries collected from 1998 to 2017. Lede-3, Oracle Fragments, Text Rank, Abstractive Sequence to sequence models and pointer-generator have been tested on this dataset [51]. Lead -10 extractor, pointer and classifier models and modified transformer language models are also applied on this dataset [25].

Turkish news dataset containing short and content news of 112,833 records of a duration of 5 years. Abstractive summarization was successfully performed on this dataset using sequence to sequence model [29].

Sindhi text corpus of 15,788 documents from online books, newspapers, magazines, blogs and social websites were collected and analyzed using TFIDF, document term matrix ready to be used for information retrieval along with other NLP tasks [52].

PubMed consists of more than 33 million records for biomedical literature from MEDLINE, life science journals, and online books. SumBasic, LexRank, LSA, Seq2Seq, PointerGen, Discourse-Aware attention model have been applied on this dataset [53]. Lead -10 extractor, pointer and classifier models and modified transformer language models are also applied on this dataset [25].

arXiv is a free distribution service and an open-access archive for 2,026,977 scholarly articles in the fields of physics, mathematics, computer science, quantitative biology, quantitative finance, statistics, electrical engineering and systems science, and economics. SumBasic, LexRank, LSA, Seq2Seq, PointerGen, Discourse-Aware attention model have been applied on this dataset [53]. Lead -10 extractor, pointer and classifier models and modified transformer language models are also applied on this dataset [25].

BigPatent dataset consists of 1.3 million records of U.S. patent documents along with human written abstractive summaries. It contains patents filed after 1971 across nine different technological areas. Lead-3, OracleFrag, OracleExt, TextRank, LexRank, SumBasic, RNN-Ext RL, Seq2Seq, PointGen, PointGen+Cov, SentRewriting have been tested on this dataset [54]. Lead -10 extractor, pointer and classifier models and modified transformer language models are also applied on this dataset [25].

Email Dataset of Enron Corporation consisting of 619,446 messages from 158 users. Available for Natural Language Processing tasks are manually created abstract summaries of maximum 450 characters, 5 most important sentences extracted and ranked, 5 important keywords or phrases identified, and emails classified into corporate or private [55]. This dataset was used in text summarizing using techniques including bag of words, word2vec, Auto-Encoder (AE), Variational Auto-Encoder (VAE) and Extreme Learning Machine Auto Encoder (ELM- AE) [26]

Clinical Trials contained 277,228 records from which 101,016 were used for extractive text summarization using LexRank, TextRank, Luhn, LSA, SumBasic and KLSum algorithms. The dataset contained detailed and brief summaries. [41]

Table 7. Summary of Datasets

Type	**Datasets**
News	Turkish Dataset, DUC, Newsroom, Sindhi Dataset
Journal, Articles, Online Books, Magazines, etc	Sindhi Dataset, PubMed, arXiv
Patent	BigPatent
Email	Enron Corporation Dataset
HealthCare	ClinicalTrials.gov

3.4 Health Care Domain

In this section we mention any Natural Language Processing work done in the Health Care domain as work done including textual summarization to get ideas and awareness of other related aspects such as the type of word embeddings used for example. Following the list, we provide a summary of the language, methods and dataset used by researchers.

Damianos et al. from the Mitre corporation write to explain the processes used by Mitap System for monitoring Bio Events. This system used methods to display pop-up keyword lists and automatic summaries. Machine learning techniques were used to extract relevant sentences. The system also included human-made summaries. They also mention the use of natural language analyzer named Alembic and the use of machine translations. The purpose to mention this paper is to let the audience realize that work related to this thesis has been commercially used as early as 2002. [2]

Suominen et al. attempt to rank extracts of nursing notes with respect to their relevance ie. Breathing, blood circulation, pain. Regularized Least squares algorithm was used for this purpose. To evaluate the results a rank correlation coefficient Kendall's τb was used to compare rankings produced by the nurses and those generated by the classifier. The dataset of 43 patients was extracted in 2001 from ICU department of a Finnish Hospital. The text was split manually into smaller pieces indicating one topic or subject. [17]

Moen et al. attempt to evaluate whether the results of assessing clinical summaries using golden summaries correlate with that of the human judgement. In their experiment, they use eight different methods including the original, random and oracle based. To evaluate, four variants of rouge score were compared with manual scores. Wilcoxon test and Spearman's rank correlation coefficient were also calculated. The dataset used for this work contained records of 26,000 inpatients from a Finnish hospital between 2005 and 2009. [19]

Morid et al. attempt to classify clinical knowledge source UpToDate articles. To produce golden summaries, 1072 sentences were rated amongst 3 clinicians with a 5-pointer scale and were tested against random set of 140 PubMed abstracts. They experimented using kernel-based Bayesian network, naïve bayes, neural network, support vector machine, k-nearest neighbour and logistic regression using several features including UML based concepts and other semantic groups and cue words. They use MedTagger to extract the UML based concepts and SemRep to extract semantic predictions. They computed average, precision,

recall and F-measure values to evaluate the best classifier. The dataset consisted of 4,824 sentences from six chronic conditions from UpToDate articles. [20]

Moen et al. attempt to summarize clinical free text notes of cardiac patients. The researchers use various data pre-processing and feature extraction methods including filtering ICD-10 code 125, using word2vec and TF-IDF techniques. Several methods most of them based on statistical values derived from the corpus including Composite, Oracle, Case-Based, Translate, Random, Repeated Sentences, Last Sentences, Centrality were applied evaluating the rouge score. Manual evaluations of 20 care episodes containing 8 summaries per care episode were also done, and Spearman's rho results were used to corelate the manual with the rouge score. The dataset used is from 66,884 care episodes of cardiac inpatients between 2005 and 2009. [18]

Moradi develops a Clustering and Itemset based Biomedical Summarizer (CIBS) to extract biomedical concepts from single and multiple documents. In the pre-processing phase, he uses the MetaMap program developed by the US National Library of Medicine which identifies noun phrases and maps it to the UML dictionary returning all the containing concepts. Using the apriori algorithm to identify the top frequent concepts, sentences were clustered according to the concepts/topics using the hierarchical clustering algorithm. CIBS then extracts the most important sentences from each cluster to cover all the concepts. CIBS with different number of clusters was evaluated using the rouge score method comparing the results with MEAD, SUMMA and TextLexAn summarizers. The dataset contained 25 collections of 300 multiple documents from PubMed with model summary using Wikipedia. For single document, 400 scientific biomedical articles from BioMed's central's corpus for text mining research was used. [21]

Moradi works on extracting meaningful text from biomedical texts. In the data cleaning process, he maps the text with standard Unified Medical Language (UML) concepts using MetaMap semantic types. The output contains multiple sentences each contains multiple concepts. Apriori algorithm was then used to select the most frequently occurring sets. From these sets, the most meaningful itemsets are selected using a meaningfulness measure which are then plotted as a graph where the vectors denote the sentences and the edges the relationship between the sentences. Consequent sentences and those that have common frequent items are related. Most important sentences are those with the highest degree and are extracted for the summary. The dataset contained 300 biomedical full-text articles from the

BioMed Central's corpus for text mining research. They compared their results using rouge score with other known algorithms such as SUMMA, MEAD and TexLexAn.[22]

Gulden et al. use extractive text summarization methods to create summaries of clinical trial descriptions available at clinicaltrials.gov. Researchers apply LexRank, TextRank, LSA, Luhn, SumBasic and KLSum algorithms using python sumy library. Using the available reference summaries, they evaluate the rouge score, precision, recall for each method include random as the baseline. They also evaluate the summaries by human evaluation method to compare the results. The data used for this work was 101,016 of 277,228 records. [41]

Korach et al. use natural language processing techniques to extract most important set of words or phrases. They tokenize and clean the documents then use FastText for embedding maximum of 4 n-grams. Then they use the TextRank algorithm on each document to extract scores of each phrase iterating this 200 times. After this, a greedy algorithm is used to extract top scored sentences. C-Value / NC-value algorithm was then used to extract the globally important n-grams. 240 phrases were manually evaluated by 2 clinicians. The data used was of Partners' Hospital's 61,740 encounters of 45,817 inpatients ranging from 2015 to 2018. This work was targeted for the hospital rapid response team. [23]

Table 8. Summary of Related work in Health Care

Article	**Language**	**Methods**	**Evaluation Method**	**Data Set Used**
Damianos et al. (2002)	English Chinese French German Italian Portuguese Russian Spanish	Natural Language Analyzer named Alembic Websumm CyberTrans	Human-made summaries	3500 to 6000 articles per day Epidemiological reports News Emails
Suominen et al. (2006)	Finnish	Regularized Least squares algorithm	Coefficient Kendall's τb was used to compare rankings produced by the nurses and those generated by the classifier.	43 patients was extracted in 2001 from ICU department of a Finnish Hospital.
Moen et al. (2014)	Finnish	Eight different methods including the original, random and oracle based	Rouge metrics with manual scores. Wilcoxon test and Spearman's rank correlation coefficient were also calculated	26,000 inpatients from a Finnish hospital between 2005 and 2009
Morid et al. (2016)	English	Kernel-based Bayesian network Naïve bayes Neural Network Support vector machine K-nearest neighbour Logistic regression	1072 sentences were rated amongst 3 clinicians with a 5-pointer scale. Scoring Mechanisms: Average Precision Recall F-measure	140 PubMed abstracts.
Moen et al. (2016)	Finnish	Composite Oracle Case-Based Translate Random Repeated Sentences Last Sentences Centrality	Rouge Metric Manual evaluations of 20 care episodes. Spearman's rho results were used to corelate the manual with the rouge score	The dataset used is from 66,884 care episodes of cardiac inpatients between 2005 and 2009.

Article	Language	Methods	Evaluation Method	Data Set Used
Moradi (2018)	English	MEAD SUMMA TextLexAn CIBS	Rouge Metrics	Multi-document corpus: 300 abstracts extracted on 25 topics each from PubMed and its summary from Wikipedia. Single-document corpus: 400 articles from BioMed Centrals' corpus for text mining research and its abstract as model summary.
Moradi (2018)	English	Graph-based Itemset based compared with SUMMA MEAD TextLexAn	Rouge Metrics	300 biomedical articles from the BioMed Central's corpus for text mining research and its abstract as model summary.
Gulden et al. (2019)	English	LexRank TextRank LSA Luhn SumBasic KLSum	Rouge Metrics Manually evaluated by 4 Reviewers (Likert scale questionnaire)	101,016 of 277,228 clinical trial descriptions registered on clinicaltrials.gov.
Korach et al. (2020)	English	FastText TextRank C-Value / NC-value algorithm	Manually evaluated by 2 clinicians.	778,955 nursing notes from Partners' Hospital's ranging from 2015 to 2018.

4 Experimental Work

From previous work, it seemed that datasets that had domain-specific vocabulary showed better results with simple statistical methods such as Bag of Words or TFIDF when compared to neural network based prediction models (Doc2Vec) or even attention models (Hierarchical Attention Network). [24]

We chose to compare a statistical based word embedding method (TFIDF) with that of a neural network based (Word2Vec). To be able to compare the results, we used the same ranking method for both models (LexRank) and took Random selection as the baseline method.

One may argue as to why TFIDF and not bag of words or why Word2Vec and not FastText for example. We chose TFIDF over bag of words because TFIDF gives a better global picture of the words as compared to simple bag of words. We chose Word2vec over FastText because FastText considers the morphological characteristics of a word and when comparing with TFIDF method which focuses on words and not the characters, Word2Vec seemed more appropriate.

In the following sections we describe the experimental process, data, methods, evaluation process and the results. In section 4.1, we briefly describe the process of the project. In section 4.2, we describe the data and mention the cleaning steps taken. In section 4.3, we describe the methods used. In section 4.4, we describe the evaluation approach chosen and in section 4.5 we discuss the results.

4.1 The Process

To gain a practical understanding of the theoretical aspects explained earlier in Section 2.2 titled Text Summarization, a small python-based project is developed.

The project aims at traversing through JSON data, pre-processing, establishing definitions of different techniques and using them on the nursing notes dataset to produce summaries. Further details of the code can be found in the Appendix Section.

The project was divided into the following steps:

1. Creation of the corpus: JSON data files were traversed to fetch the notes, send for data cleaning and then create a corpus.
2. Building the model: Models learned from the training dataset and saved.
3. Testing the model: Models were run on pre-processed testing data.
4. Formatted Results: Results were formatted and saved in excel file.

A substantial amount of time was spent exploring the data and searching for ideal summaries to build a self-evaluating, self-learning system. However, since golden summaries were not found, evaluating results were done manually by two volunteers. Their feedbacks are described and discussed in 4.4 - 4.5.

4.2 The Data

The data received was already de-identified to preserve confidentiality and prevent linking any information to the actual patient. Permission to work on the data was taken from the hospital and ethical approval was obtained.

The data gathered and sorted by the research team was in JSON format found in two folders: the physician data and the nursing data. The physician folder contained inpatient records from 67,487 patients. The nursing folder had four folders containing 13,973 inpatient records for "Care Acquity", 65,384 inpatient records for "Care Notes", 58,693 inpatient records for "Care Tables" and 10,080 inpatient records for "HOI text".

The physician folder was used to obtain inpatients' episodes. An episode is the duration of a patient's stay at the hospital usually containing the start date as the day the patient was admitted in the hospital and an end date as the day the patient was discharged from the hospital. From a total of 4,639,521 episodes, we found 846,371 episodes that had values for both the admission date and the discharge date. The rest of the episodes were discarded.

The 846,731 episode ranges obtained from the physician dataset was then used to extract the nursing notes corresponding to each episodes' duration. We found a total of 614,090 episodes in the nursing folder: 13,368 episodes for "Care Acquity", 400,029 episodes for "Care Notes", 200,098 episodes for "Care Tables" and 595 for "HOI Text".

Each folder contained multiple notes for a particular episode range consisting of *Timestamp* to indicate when the note was taken, *Metadata* to give additional information about the note such as which ward or which unit the note was taken in and the *Content* which is the note itself.

The table below shows the type of metadata information found against each folder:

Table 9. The Data – Nursing Notes

Folders	**Meta Data**
Care Acquity	Hospital unit code, Ward, Author, Care Process Phase, Header, Headers List, CA Class, CA Points
Care Notes	Hospital unit code, Ward, Entry View, Care Process Phase, Headers List
Care Tables	Hospital unit, Ward
HOI Text	Hospital Unit Code, Ward, Entry View, Care Episode Label, Care Episode Start, Care Episode End, Note Category

For the scope of this experimental work, we will focus only on “Care Notes” as it is the only one containing unstructured data. Ideally the summaries should consist of both the structured and unstructured data that would require specialized algorithm for each data type. For simplicity, in this thesis we aim at the initial phase of addressing the unstructured data and hence we leave the work for structured and combined summarization for future work.

Pre-processing of the data included tokenizing, converting text to lowercase, removing punctuation marks and stop words.

Tokenizing: After using the sent_tokenize function of nltk library, we then used the regex expression for carriage return (\r) and new line (\n).

Stop word: To the nltk stopwords standard dictionary of words we added some commonly found words specific to our text such as 'teksti', 'otsikko', 'mg', 'iv', 'klo', 'mmhg', 'ml'.

4.3 Methods

For this experimental work, we have three models:

1. Random method as the baseline model which extracts sentences based on chance and collects them to produce the summary.
2. Word2Vec_LexRank which uses Word2Vec model[44], [45] for word embedding and then the Lex Rank algorithm [47] for the selection of sentences.
3. TFIDF_Lexrank which used the text feature Term Frequency – Inverse Document Frequency [42], [43] to generate vectors and then with the Lex Ranking algorithm [47] extracted sentences for the summary.

Word2Vec_LexRank method and TFIDF_Lexrank method work in a similar way with the difference in the featured model used for executing test runs. Both methods perform the following main steps:

1. Tokenize the original nursing notes (containing meta data) and give each sentence a sequential number for reference.
2. Compile all the sentences in one string and clean the data
3. Compute sentence vectors.
4. Derive the similarity matrix and form the graph.
5. Using the Lex ranking algorithm, compute the scores of each sentence.
6. Extract top 30 sentences mapping them to the original data containing the meta data.

4.4 Evaluation

To evaluate the results, 15 episodes shown in the table below were given as an input to the three models that produced summaries for each method. Episodes with more than 10 notes were selected. The summaries along with the original text were extracted in an excel sheet to be graded by two human beings.

Table 10. Sample Data of 15 Episodes

Episodes	Patient	Year	Length of Stay	No. of Notes	No. of sentences
1	1	2011	6	155	465
2	1	2017	1	29	234
3	1	2017	1	40	222
4	2	2018	1	32	300
5	2	2018	7	64	556
6	3	2016	14	333	1657
7	4	2015	1	19	103
8	5	2016	1	18	111
9	6	2015	1	10	59
10	6	2019	1	12	59
11	7	2016	3	54	243
12	8	2016	4	38	143
13	8	2020	1	10	77
14	9	2019	1	14	79
15	9	2019	2	45	203

As can be seen from the table above, we have data ranging from 2011 to 2020, mostly with a length of stay of 1 and an average of 300 sentences per episode. Some episodes are of the same patient.

The grading scale provided to evaluators was a number from 1 – 4 with the following evaluating criteria:

1. The summary gives a good overview of the care episode.
2. The summary gives a mediocre overview of the care episode.
3. The summary gives a poor overview of the care episode.
4. Unable to assess the summary.

4.5 Results

The table below and the graph following it shows the summary of the results from the evaluators.

Table 11. Overview of Evaluator Results

	Random		**Word2Vec_LexRank**		**TFIDF_LexRank**	
Episodes	Evaluator 1	Evaluator 2	Evaluator 1	Evaluator 2	Evaluator 1	Evaluator 2
1	3	4	2	2	3	3
2	2	4	2	2	3	2
3	2	4	2	2	2	3
4	3	4	2	3	2	3
5	2	4	3	3	3	3
6	3	3	2	4	3	3
7	3	4	3	4	3	4
8	3	3	3	3	3	3
9	3	4	3	3	2	3
10	2	4	1	1	1	1
11	3	4	2	3	2	3
12	3	3	2	2	2	2
13	3	4	2	2	3	4
14	3	3	1	1	2	1
15	3	3	2	2	2	2

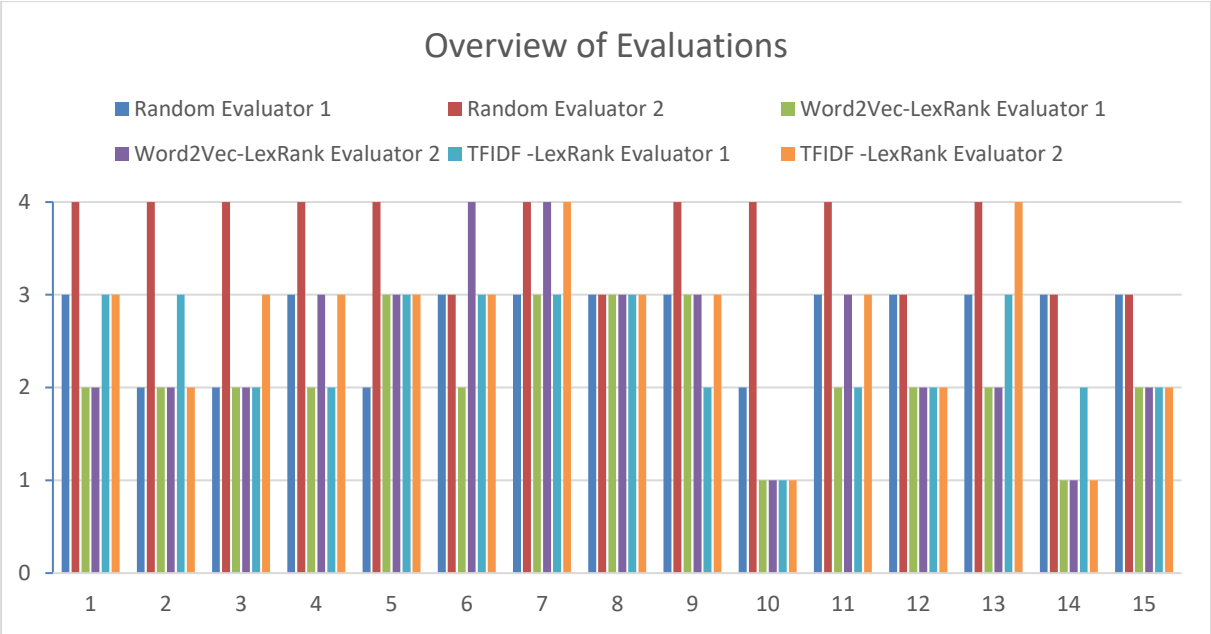

Figure 4.Overview of Evaluator Results

From the table and graph above, we can see that for the Random selection, none of the evaluators rate any summary as good. We see that both the evaluators are in agreement with each other for 5 episodes. For episodes number 6, 8, 12, 14 and 15 both evaluators rate these as a poor summary. For episodes 2, 3, 5 and 10 one evaluator rates them as mediocre summaries and rates poor for episodes 1, 4, 7, 9, 11, 13 whilst the second evaluator fails to assess any of them.

For Word2Vec_LexRank model, we see the best agreement between both evaluators as compared to Random and TFIDF_Lexrank with 11 similar ratings. For episodes 10 and 14, both evaluators believe the summaries produced by the system were good. For episodes 1, 2, 3, 12, 13 and 15 both the evaluators rate the summaries as mediocre. For episodes 5, 8 and 9 both evaluators believe the summaries were poor. For episode 4 and 11 one evaluator rates them as mediocre whilst the other one rates it as a poor overview of the episode. Evaluator one rates the summary for episode 6 as a mediocre summary and for episode 7 as a poor one whilst evaluator two fails to assess both of them.

For TFIDF_Lexrank model, we see that both the evaluators are in agreement with each other over 7 episodes. Both evaluators believe the summary for episode 10 was good, the summary for episodes 12 and 15 were mediocre and summary for episodes 1, 5, 6 and 8 were poor. For episodes 3, 4, 9 and 11 evaluator one rated them as mediocre whilst evaluator two thought of them as a poor overview. For episode 7 and 14, evaluator one rated the summaries as poor

whilst the second evaluator was unable to assess. For episode 14, evaluator one believed the summary was mediocre but evaluator two rated it as good. For episode 2, evaluator one rated it as poor whilst evaluator 2 believed it to be mediocre.

We can see, also from the Tables 10-11 that both evaluators have given higher ranks to Word2Vec_LexRank and TFIDF_LexRank as compared to the Random method. We also see that both evaluators believe Word2Vec_Lexrank performed better than TFIDF_LexRank as they have graded ranks 1 and 2 for more episodes in Word2Vec_LexRank than the TFIDF_LexRank method.

Table 12. Overview of Results from Evaluator 1

Evaluator 1				
Grades	**1**	**2**	**3**	**4**
Random		4	11	
Word2Vec_LexRank	2	9	4	
TFIDF_LexRank	1	7	7	
Total	3	20	22	0

Table 13. Overview of Results from Evaluator 2

Evaluator 2				
Grades	**1**	**2**	**3**	**4**
Random			5	10
Word2Vec_LexRank	2	6	5	2
TFIDF_LexRank	2	3	8	2
Total	4	9	18	14

Although the sample size is very small to be able to provide strong support from statistical tests, we compute Wilcoxon signed-rank test. Comparing the baseline random method with word2vec gave a score of 6.0 and p-value of 0.023 and with TFIDF a score of 13.5 and p-value of 0.145. Comparing Word2Vec and TFIDF, a score of 2.5 and a p-value of 0.046 is obtained.

Interpreting the results, we do not have enough evidence to conclude a difference in the results between TFIDF and Random method, however, we can say the results of word2vec

when compared to random method are different in a statistically significant manner and the results of word2vec when compared to TFIDF are different in a statistically significant manner.

To assess the correlation between the evaluator's results, we compute kendalltau test for each method. For the baseline random method, we obtain a tau value of -0.42 and p-value of 0.111, for Word2Vec, we obtain a tau value of 0.705 and p-value of 0.003 and for TFIDF, we obtain a tau value of 0.535 and a p-value of 0.028.

Statistically we can say that both the evaluators' results are highly corelated for TFIDF and Word2vec methods but not for the Random method.

Other than the gradings, evaluators also shared their valuable comments. Some of them are as following:

- A lot of headings, little actual text.
- Repetition in the data hence repetition in summaries.
- Mentioned once but important information found missing.

Data Cleaning process could be revisited and headings without or very little text could be discarded. A mechanism could be built to check repeated sentences and exclude them from the final summaries.

After reading all the comments from the evaluators, it seems they are seeking answers to some questions like why the patient was there, which surgical treatment was given to the patient etc. Perhaps experimenting with transformer-based models that are based on queries and keys may give better results for extracting valuable information.

From previous literature it has been noted that Word2vec generally performs better that TFIDF for semantically classifying text [56], [57] however we also find some study which shows that simple statistical models outperform complex models specifically in cases that have domain-specific vocabularies [24]. As a summary, due to the small sample size we cannot accept or reject our null hypothesis however it seems that word2vec can show promising results for healthcare nursing notes. In future we will increase the sample size to provide strong statistical evidence.

5 Closure

5.1 Limitations

The idea of building a text summarizer is challenging yet fascinating. The learning process was significant, however, lack of previous knowledge in the subject area required some time and efforts to understand even the basic concepts. Since this was a new subject area for the author and the time was limited, the work of this thesis was at of a beginner's level - new revolutionary techniques like transformer-based models (eg. BERT, GPT-2, XLNet) were not explored.

Although it seemed when building machine models, knowledge of the human knowledge is not essential as the machine works on numbers, the author of this work feels that knowing the language would have had an added advantage. Specifically in the phases of analysing the initial data input and the final summary output. Working closely with someone having the knowledge could be a solution or designing models first for a similar dataset of known language and then implementing on foreign languages could be another possibility.

To deliver a solution to be used by end users, a close communication and requirement analysis is essential. However, due to cultural differences, sensitivity of information from hospital data, partly due to the covid pandemic and my lacking experience, the work on this thesis was not done with frequent communications with the users.

Although four different subsets of nursing folders were explored, only the Care Notes (unstructured data) was focused on for this thesis. In future, the data in other folders could also be addressed.

5.2 Further Work

For simplicity, multiple notes of one episode were compiled into one large text used for summarization. This lost the information such as which note number it belonged to or what time the text was recorded. Perhaps, we may want to first assess which note is important and give it a higher weight than the other notes.

Although in this work, we had limited the number of notes to more than 10, perhaps a more interesting criteria would be a longer range of episodes and compiled notes containing at least 300 - 500 sentences.

Clinical or nursing text have a special dictionary of words, maybe they can be given a different special treatment like specifically including sentences mentioning a particular summary for example.

Due to the limitation of time, the sample episodes were too small to draw any conclusion for the whole dataset, thus in further work the data used for evaluating can be larger.

5.3 Conclusion

In this thesis, we discuss the theoretical background concepts required to briefly understand the broader domain of Natural Language Processing and then detailed concepts of Text Summarization and then we discuss Health Care.

This document then includes related relevant and interesting work done by other publishers. The papers included were mostly related to the methodologies used in the practical implementation but also included any text summarization work done in the health domain.

Also, as a part of this thesis we develop a python-based module to summarize finnish nursing notes and evaluate the performance between random selection, Word2Vec_Lexrank and TFIDF_Lexrank algorithms.

Our initial first hypothesis was that both Word2vec_Lexrank and TFIDF_Lexrank algorithm will perform better than the baseline random selection. Our second hypothesis statement was that Word2vec_Lexrank algorithm would perform better than TFIDF_Lexrank. From our results and discussions, we can conclude that both Word2Vec_Lexrank and TFIDF_Lexrank algorithms perform better than the random selection which was our first assumption and as expected Word2vec_Lexrank performed better than TFIDF_Lexrank.

This thesis also includes an appendix which documents the code. Future work and limitations are discussed in the previous section.

References

[1] K. Wisner, A. Lyndon, and C. A. Chesla, "The electronic health record's impact on nurses' cognitive work: An integrative review," *International Journal of Nursing Studies*, vol. 94, pp. 74–84, 2019, doi: 10.1016/j.ijnurstu.2019.03.003.

[2] L. Damianos *et al.*, "Real Users, Real Data, Real Problems: The MiTAP System for Monitoring Bio Events," *Proceedings of the Second International Conference on Human Language Technology Research*, pp. 357–362, 2002, [Online]. Available: https://dl.acm.org/doi/abs/10.5555/1289189.1289227

[3] A. Goldstein and Y. Shahar, "An automated knowledge-based textual summarization system for longitudinal, multivariate clinical data," *Journal of Biomedical Informatics*, vol. 61, pp. 159–175, Jun. 2016, doi: 10.1016/J.JBI.2016.03.022.

[4] A. Bradford, "The Five (and More) Senses," *Live Science*, 2017. https://www.livescience.com/60752-human-senses.html (accessed Dec. 17, 2021).

[5] "Languages used on the Internet - Wikipedia," *En.wikipedia.org*, 2021. https://en.wikipedia.org/wiki/Languages_used_on_the_Internet (accessed Sep. 18, 2021).

[6] H. P. Luhn, "The Automatic Creation of Literature Abstracts," *IBM Journal of Research and Development*, vol. 2, no. 2, pp. 159–165, Apr. 1958, doi: 10.1147/rd.22.0159.

[7] H. P. Edmundson, "New Methods in Automatic Extracting," *Journal of the ACM (JACM)*, vol. 16, no. 2, pp. 264–285, 1969, doi: 10.1145/321510.321519.

[8] "Cosine similarity," *En.wikipedia.org*, 2021. https://en.wikipedia.org/wiki/Cosine_similarity

[9] R. Mihalcea and P. Tarau, "TextRank: Bringing order into texts," in *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, 2004, pp. 404–411. [Online]. Available: https://aclanthology.org/W04-3252

[10] "What information does an electronic health record (EHR) contain? | HealthIT.gov," *Healthit.gov*, 2021. https://www.healthit.gov/faq/what-information-does-electronic-health-record-ehr-contain (accessed Dec. 22, 2021).

[11] J. C. Feblowitz, A. Wright, H. Singh, L. Samal, and D. F. Sittig, "Summarization of clinical information: A conceptual model," *Journal of Biomedical Informatics*, vol. 44, no. 4, pp. 688–699, Aug. 2011, doi: 10.1016/j.jbi.2011.03.008.

[12] R. Pivovarov and N. Elhadad, "Automated methods for the summarization of electronic health records," *Journal of the American Medical Informatics Association*, vol. 22, no. 5, pp. 938–947, Sep. 2015, doi: 10.1093/jamia/ocv032.

[13] "Sosiaali- ja terveysministeriön asetus… 298/2009 - Säädökset alkuperäisinä - FINLEX ®," *Finlex.fi*, 2021. https://finlex.fi/fi/laki/alkup/2009/20090298#Pidm45237816097952 (accessed Dec. 22, 2021).

[14] A. Thoroddsen, A. Ehrenberg, W. Sermeus, and K. Saranto, “A survey of nursing documentation, terminologies and standards in European countries.,” *NI 2012: 11th International Congress on Nursing Informatics, June 23-27, 2012, Montreal, Canada.*, vol. 2012, p. 406, 2012.

[15] A. Avasthi, A. Ghosh, S. Sarkar, and S. Grover, “Ethics in medical research: General principles with special reference to psychiatry research,” *Indian Journal of Psychiatry*, vol. 55, no. 1, p. 86, 2013, doi: 10.4103/0019-5545.105525.

[16] K. Page, “The four principles: Can they be measured and do they predict ethical decision making?,” 2012. [Online]. Available: http://www.biomedcentral.com/1472-6939/13/1/10

[17] H. Suominen *et al.*, “Relevance Ranking of Intensive Care Nursing Narratives,” in *International Conference on Knowledge-Based and Intelligent Information and Engineering Systems*, 2006, pp. 720–727. doi: 10.1007/11892960_87.

[18] H. Moen *et al.*, “Comparison of automatic summarisation methods for clinical free text notes,” *Artificial Intelligence in Medicine*, vol. 67, pp. 25–37, 2016, doi: 10.1016/j.artmed.2016.01.003.

[19] H. Moen *et al.*, “On evaluation of automatically generated clinical discharge summaries,” in *CEUR Workshop Proceedings*, 2014, vol. 1251, pp. 101–114.

[20] M. A. Morid, M. Fiszman, K. Raja, S. R. Jonnalagadda, and G. del Fiol, “Classification of clinically useful sentences in clinical evidence resources,” *Journal of Biomedical Informatics*, vol. 60, pp. 14–22, Apr. 2016, doi: 10.1016/j.jbi.2016.01.003.

[21] M. Moradi, “CIBS: A biomedical text summarizer using topic-based sentence clustering,” *Journal of Biomedical Informatics*, vol. 88, pp. 53–61, Dec. 2018, doi: 10.1016/j.jbi.2018.11.006.

[22] M. Moradi, “Frequent Itemsets as Meaningful Events in Graphs for Summarizing Biomedical Texts,” in *2018 8th International Conference on Computer and Knowledge Engineering (ICCKE)*, 2018, pp. 135–140. doi: 10.1109/ICCKE.2018.8566651.

[23] Z. T. Korach *et al.*, “Mining clinical phrases from nursing notes to discover risk factors of patient deterioration,” *International Journal of Medical Informatics*, vol. 135, p. 104053, Mar. 2020, doi: 10.1016/j.ijmedinf.2019.104053.

[24] A. Wawrzyński and J. Szymański, “Study of Statistical Text Representation Methods for Performance Improvement of a Hierarchical Attention Network,” *Applied Sciences*, vol. 11, no. 13, p. 6113, Jun. 2021, doi: 10.3390/app11136113.

[25] J. Pilault, R. Li, S. Subramanian, and C. Pal, “On Extractive and Abstractive Neural Document Summarization with Transformer Language Models,” in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Sep. 2020, pp. 9308–9319. doi: 10.18653/v1/2020.emnlp-main.748.

[26] N. Alami, M. Meknassi, and N. En-nahnahi, “Enhancing unsupervised neural networks based text summarization with word embedding and ensemble learning,” *Expert Systems with Applications*, vol. 123, pp. 195–211, 2019, doi: 10.1016/j.eswa.2019.01.037.

[27] P. Fecht, S. Blank, and H. P. Zorn, “Sequential transfer learning in NLP for German text summarization,” 2019.

[28] M. Lee, “An empirical evaluation of models of text document similarity,” 2005. [Online]. Available: http://digital.library.adelaide.edu.au/dspace/handle/2440/28910

[29] F. Ertam and G. Aydin, “Abstractive text summarization using deep learning with a new Turkish summarization benchmark dataset,” *Concurrency Computation* , no. May, pp. 1–10, 2021, doi: 10.1002/cpe.6482.

[30] E. Heidary, H. Parvïn, S. Nejatian, K. Bagherifard, and V. Rezaie, “Automatic Persian Text Summarization Using Linguistic Features from Text Structure Analysis,” *Computers, Materials and Continua*, vol. 69, no. 3, pp. 2845–2861, 2021, doi: 10.32604/cmc.2021.014361.

[31] A. Alzuhair and M. Al-Dhelaan, “An Approach for Combining Multiple Weighting Schemes and Ranking Methods in Graph-Based Multi-Document Summarization,” *IEEE Access*, vol. 7, pp. 120375–120386, 2019, doi: 10.1109/ACCESS.2019.2936832.

[32] S. Narayan, S. B. Cohen, and M. Lapata, “Ranking sentences for extractive summarization with reinforcement learning,” *NAACL HLT 2018 - 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, vol. 1, pp. 1747–1759, Feb. 2018, doi: 10.18653/v1/n18-1158.

[33] J. M. Sanchez-Gomez, M. A. Vega-Rodríguez, and C. J. Pérez, “Extractive multi-document text summarization using a multi-objective artificial bee colony optimization approach,” *Knowledge-Based Systems*, vol. 159, pp. 1–8, Nov. 2018, doi: 10.1016/j.knosys.2017.11.029.

[34] H. van Lierde and T. W. S. Chow, “Learning with fuzzy hypergraphs: A topical approach to query-oriented text summarization,” *Information Sciences*, vol. 496, pp. 212–224, Sep. 2019, doi: 10.1016/j.ins.2019.05.020.

[35] P. Verma and H. Om, “MCRMR: Maximum coverage and relevancy with minimal redundancy based multi-document summarization,” *Expert Systems with Applications*, vol. 120, pp. 43–56, Apr. 2019, doi: 10.1016/j.eswa.2018.11.022.

[36] P. Verma, S. Pal, and H. Om, “A Comparative Analysis on Hindi and English Extractive Text Summarization,” *ACM Transactions on Asian and Low-Resource Language Information Processing*, vol. 18, no. 3, pp. 1–39, Jul. 2019, doi: 10.1145/3308754.

[37] R. Khan, Y. Qian, and S. Naeem, “Extractive based Text Summarization Using KMeans and TF-IDF,” *International Journal of Information Engineering and Electronic Business*, vol. 11, no. 3, pp. 33–44, May 2019, doi: 10.5815/ijieeb.2019.03.05.

[38] C. Fang, D. Mu, Z. Deng, and Z. Wu, "Word-sentence co-ranking for automatic extractive text summarization," *Expert Systems with Applications*, vol. 72, pp. 189–195, 2017, doi: 10.1016/j.eswa.2016.12.021.

[39] K. Shetty and J. S. Kallimani, "Automatic extractive text summarization using K-means clustering," in *2017 International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT)*, 2017, vol. 2018-Janua, no. 6, pp. 1–9. doi: 10.1109/ICEECCOT.2017.8284627.

[40] R.-M. Botarleanu, M. Dascalu, L. K. Allen, S. A. Crossley, and D. S. McNamara, "Automated Summary Scoring with ReaderBench," in *International Conference on Intelligent Tutoring Systems*, 2021, pp. 321–332. doi: 10.1007/978-3-030-80421-3_35.

[41] C. Gulden *et al.*, "Extractive summarization of clinical trial descriptions," *International Journal of Medical Informatics*, vol. 129, pp. 114–121, 2019, doi: 10.1016/j.ijmedinf.2019.05.019.

[42] K. Sparck Jones, "A Statistical Interpretation of Term Specificity and its Application in Retrival," *Journal of Documentation*, vol. 28, no. 1, pp. 11–21, 1972, doi: 10.1108/eb026526.

[43] K. S. Jones, "Index term weighting," *Information Storage and Retrieval*, vol. 9, no. 11, pp. 619–633, Nov. 1973, doi: 10.1016/0020-0271(73)90043-0.

[44] T. Mikolov, W. T. Yih, and G. Zweig, "Linguistic Regularities in Continuous Space Word Representations," 2013.

[45] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," *arXiv preprint arXiv:1301.3781*, 2013.

[46] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching Word Vectors with Subword Information," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, Dec. 2017, doi: 10.1162/tacl_a_00051.

[47] G. Erkan and D. R. Radev, "LexRank: Graph-based Lexical Centrality as Salience in Text Summarization," *Journal of Artificial Intelligence Research*, vol. 22, pp. 457–479, Dec. 2004, doi: 10.1613/jair.1523.

[48] L. Page, S. Brin, R. Motwani, and T. Winograd, "The PageRank citation ranking: Bringing order to the web.," *Stanford InfoLab*, Nov. 1999.

[49] C.-Y. Lin, "ROUGE: A Package for Automatic Evaluation of Summaries," in *Text Summarization Branches Out*, 2004, pp. 74–81.

[50] H. Olmos, S. Gómez, M. Alcañiz, M. Contero, M. P. Andrés-Sebastiá, and N. Martín-Dorta, "Design for Teaching and Learning in a Networked World," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2015, vol. 9307, pp. 613–616. doi: 10.1007/978-3-319-24258-3.

[51] M. Grusky, M. Naaman, and Y. Artzi, "Newsroom: A Dataset of 1.3 Million Summaries with Diverse Extractive Strategies," in *Proceedings of the 2018 Conference of the North American*

Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), 2018, vol. 1, pp. 708–719. doi: 10.18653/v1/N18-1065.

[52] M. A. Dootio and A. I. Wagan, "Development of Sindhi text corpus," *Journal of King Saud University - Computer and Information Sciences*, vol. 33, no. 4, pp. 468–475, May 2021, doi: 10.1016/j.jksuci.2019.02.002.

[53] A. Cohan *et al.*, "A Discourse-Aware Attention Model for Abstractive Summarization of Long Documents," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, 2018, vol. 2, pp. 615–621. doi: 10.18653/v1/N18-2097.

[54] E. Sharma, C. Li, and L. Wang, "BIGPATENT: A Large-Scale Dataset for Abstractive and Coherent Summarization," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 2204–2213. doi: 10.18653/v1/P19-1212.

[55] V. Loza, S. Lahiri, R. Mihalcea, and P. H. Lai, "Building a dataset for summarization and keyword extraction from emails," in *Proceedings of the 9th International Conference on Language Resources and Evaluation, LREC 2014*, 2014, pp. 2441–2446.

[56] M. Arora, V. Mittal, and P. Aggarwal, "Enactment of tf-idf and word2vec on Text Categorization," in *Proceedings of 3rd International Conference on Computing Informatics and Networks*, 2021, vol. 167, pp. 199–209. doi: 10.1007/978-981-15-9712-1_17.

[57] D. Rahmawati and M. L. Khodra, "Word2vec semantic representation in multilabel classification for Indonesian news article," in *2016 International Conference On Advanced Informatics: Concepts, Theory And Application (ICAICTA)*, Aug. 2016, pp. 1–6. doi: 10.1109/ICAICTA.2016.7803115.

Appendix 1 Code Document

The purpose of this document is to give an overview of the code written for this thesis.

1.1 Project Overview

The main objective of this application is to produce summaries of nursing documents using different techniques. For the scope of this project, the models included are:

1- Random Method,

2- Word2Vec with Lex Ranking,

3- TF-IDF with Lex Ranking

To achieve this, the project has the following main functionalities:

1. Data Formatting: Iterating through folders, extract nursing notes from JSON files and modify it to the required format (corpus file or list)
2. Model Training: For this project, Word2vec and TFIDF were trained
3. Testing: Running the random method and trained model methods on data

The complete code for this project can be found in the ftproject folder which can also be called the root of the project. The code is structured such that it can be reused later for additional models. Function names and variables have been chosen carefully so they are easily understood. The root contains a setup file and six sub folders described below. The detailed explanation of the files including the setup file can be found later in this document.

1. Data: Contains python script to create a corpus, the corpus itself and log of the files used to create the corpus.
2. Model: Contains python script to train models and the model files.
3. Main: Contains code iterating through data, running the test on models and code to write the output file in excel format.
4. Result: Contains the summary file(s).
5. Shared: Contains data cleaning, nursing and physician common code used by code in Data and Main folder. This folder also has the config.ini file.
6. Utilities: Contains simple functions that are shared across the project such as writing to file.

Following diagram shows an overview of how files and functions are connected to achieve the above requirements.

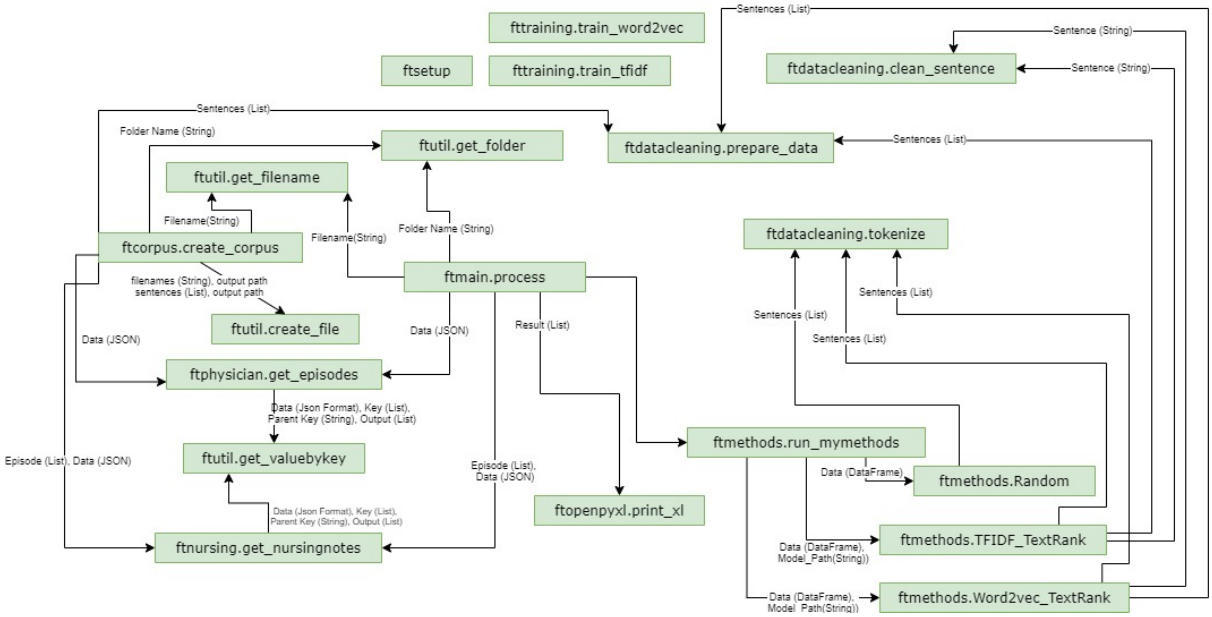

Figure 5. Diagrammatic Overview of the project

1.2 Setup

ftsetup.py file can be found at the root of project (ftproject). The file can be run as:

```
fartaj@ikitik:~/ftproject$ python3 ftsetup.py
```

Figure 6. Running ftsetup.py

Purpose: To generate the config.ini file being used in the whole project.

Usage: Make sure config.ini file does not exist in the shared folder (ftproject/shared). If config.ini exists, delete the file. Make the required changes in ftsetup.py and save the file. Enter the root folder from the terminal and simply run the setup.py file by using the command python3 ftsetup.py. You should get “Setup completed.” message if successful. If config.ini file already exists, it will notify with “File already exists, please delete and re-run.”. In case of any unexpected error it will give “Oops.. Failed to update.”

1.3 Configuration settings for this project:

Config.ini can be found in ftproject/shared folder. The file mainly contains information for the location of the JSON data, the paths of internal project and methods to be executed. Config.ini can be changed manually but best if done using the ftsetup.py file.

```
[PATHINFO]
projectroot = /home/fartaj/ftproject
nursingroot = /data/new-ikitik-data/jsondumps/IKITIK_jsonPerTable_dump/nursing/
physicianroot = /data/new-ikitik-data/jsondumps/IKITIK_jsonPerTable_dump/physician/texts

[CORPUS]
corpuslog = /home/fartaj/ftproject/data/myfiles.txt
corpusfile = /home/fartaj/ftproject/data/corpus.txt

[MODEL]
word2vecbin = /home/fartaj/ftproject/model/wordvectors.bin
word2vectxt = /home/fartaj/ftproject/model/wordvectors.txt
vectorizerpkl = /home/fartaj/ftproject/model/vectorizer.pkl

[RESULTS]
result = /home/fartaj/ftproject/results/summaries.xlsx
evaluatedsummaries = /home/fartaj/ftproject/results/evaluated.xlsx

[MYMETHODS]
method1 = Random
method2 = TextRank
method3 = TFIDF
```

Figure 7. Configuration settings

[MYMETHODS]

#The program is configured to run only those methods mentioned here. The purpose of this is to allow the application to run only specific method(s). So, if you don't want any method to run, comment that line. Please be careful not to change the name of the methods.

1.4 Corpus Creation

Create_corpus is a function found in ftproject/data/ftcorpus.py. The purpose of this function is to create a single corpus of sentences. The corpus is later used when building training models. ftcorpus.py has only one function and can be called when running the file as:

```
fartaj@ikitik:~/ftproject/data$ python3 ftcorpus.py
```

Figure 8. Running ftcorpus.py

Using the paths defined in config.ini file, the function iterates through each patient file in the physician folder and gets the episodes with the help of ftphysician.get_episodes function. For each episode, it iterates through each patient file in the four nursing folders and gets the nursing notes using ftnursing.get_nursingnotes. The function then combines each note and cleans the text using the ftdatacleaning.prepare_data function.

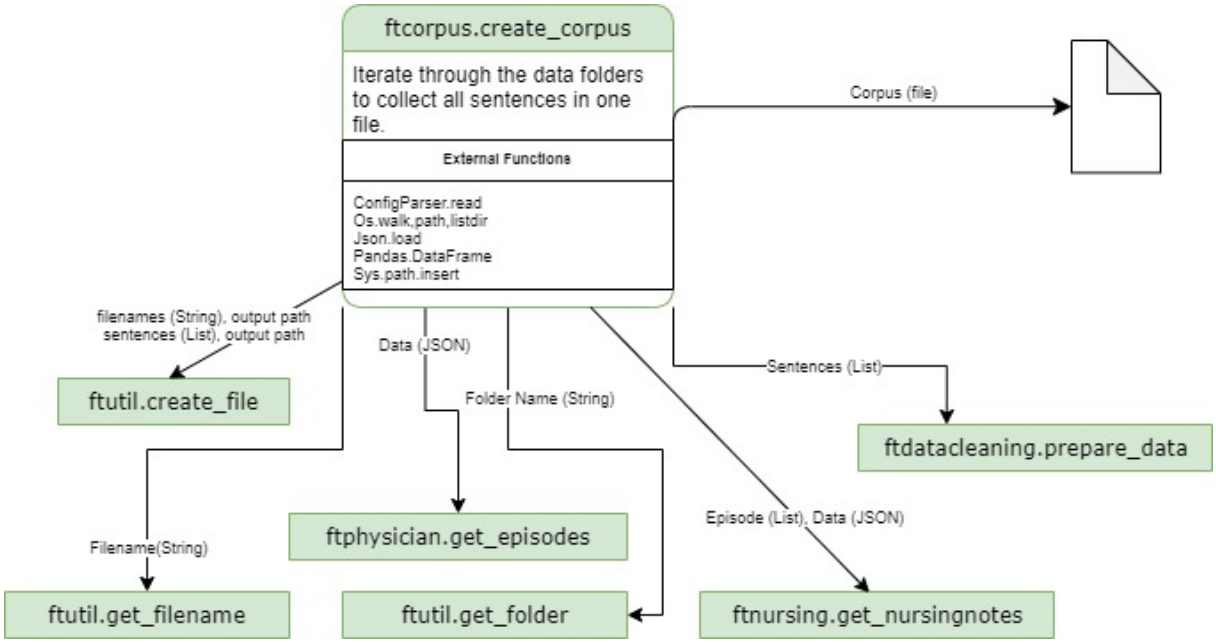

Figure 9. Generating the Corpus file

The output of this function is a log file and the corpus file. The path of both files is dynamically configurable in the config.ini file. The log file consists of all the names of the patient files the function iterates through, and the corpus file contains all the sentences. A glimpse of the file is as following:

```
1 suure cthyd virtsamäärä havainto selite
2 suure ctcir pulssi havainto selite suure ctcir verenpaine sys havainto selite
3 suure cthyd nesteet
4 havainto selite rst suure ctgen verensokeri havainto selite suure ctcir pulssi havainto selite suure ctcir
  verenpaine sys havainto selite
5 suure cthyd virtsamäärä havainto selite
6 suure ctcom lämpö havainto c selite suure ctcir verenpaine sys havainto selite suure ctcir pulssi havainto
  selite
7 suure ctcir verenpaine sys havainto selite suure ctcir pulssi havainto selite
8 suure cthyd virtsamäärä havainto selite desiduaali
9 suure cthyd nesteet
10 havainto selite nacl
```

Figure 10. Sample from Corpus file

1.5 Model Training

In ftproject/model, a file named fttraining.py is a script responsible for executing the training for models. It has two main functions; train_word2vec and train_tfidf. These two functions are called from another function named train_models and when executing this script, both the models are trained. This file can be run as:

```
fartaj@ikitik:~/ftproject/model$ python3 fttraining.py
```

Figure 11. Running fttraining.py

For both the functions the saving locations for output files and the input corpus file is read from the paths defined in the config.ini.

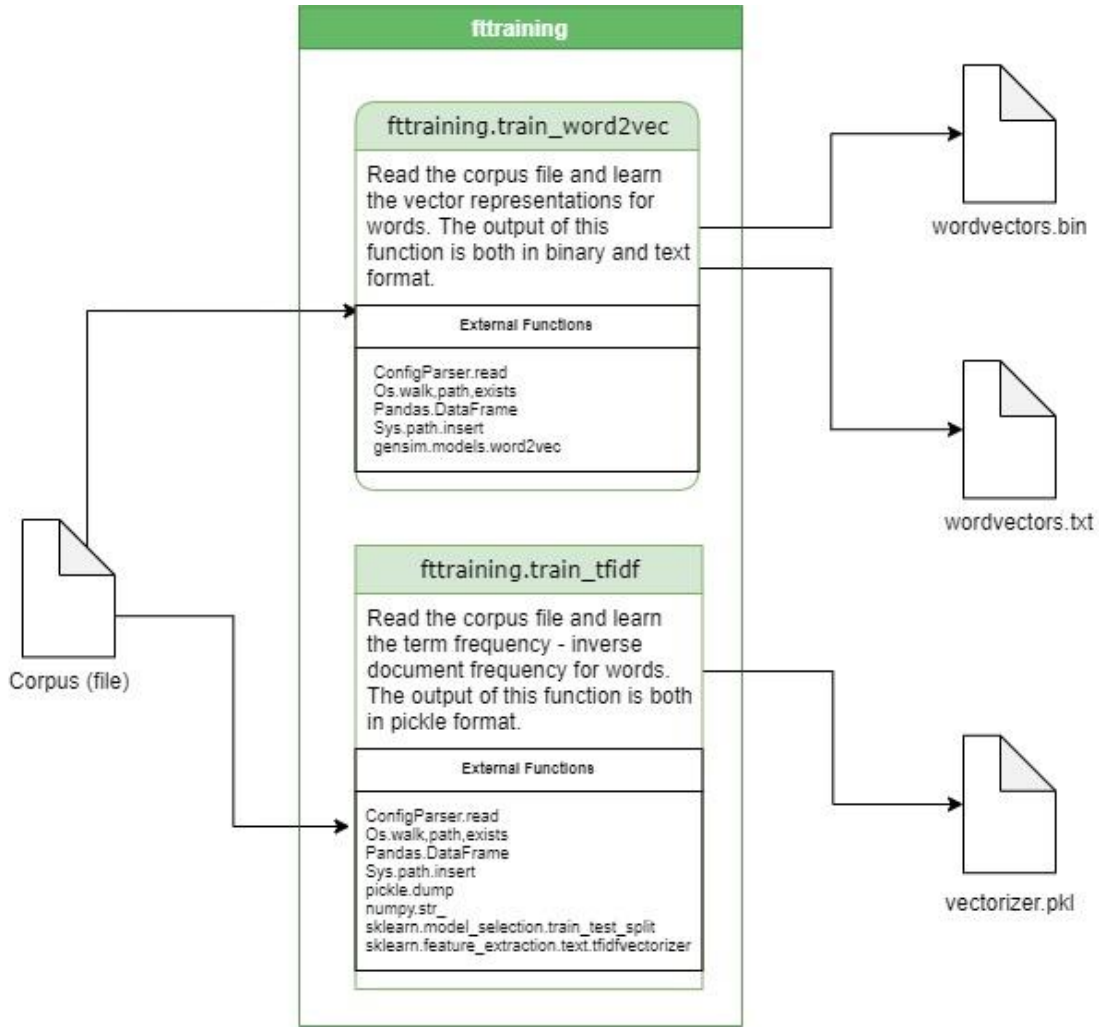

Figure 12. Training Word2vec and TF-IDF

1.6 Testing and Saving Formatted Results

In ftproject/main folder, there are three files namely ftmain.py, ftmethods.py and ftopenpyxl.py. ftmain.py is responsible for iterating through the data folders and collecting nursing notes in a list. ftmethods.py is responsible for running the trained models on the test dataset and ftopenpyxl.py is responsible to format the output and save the excel file.

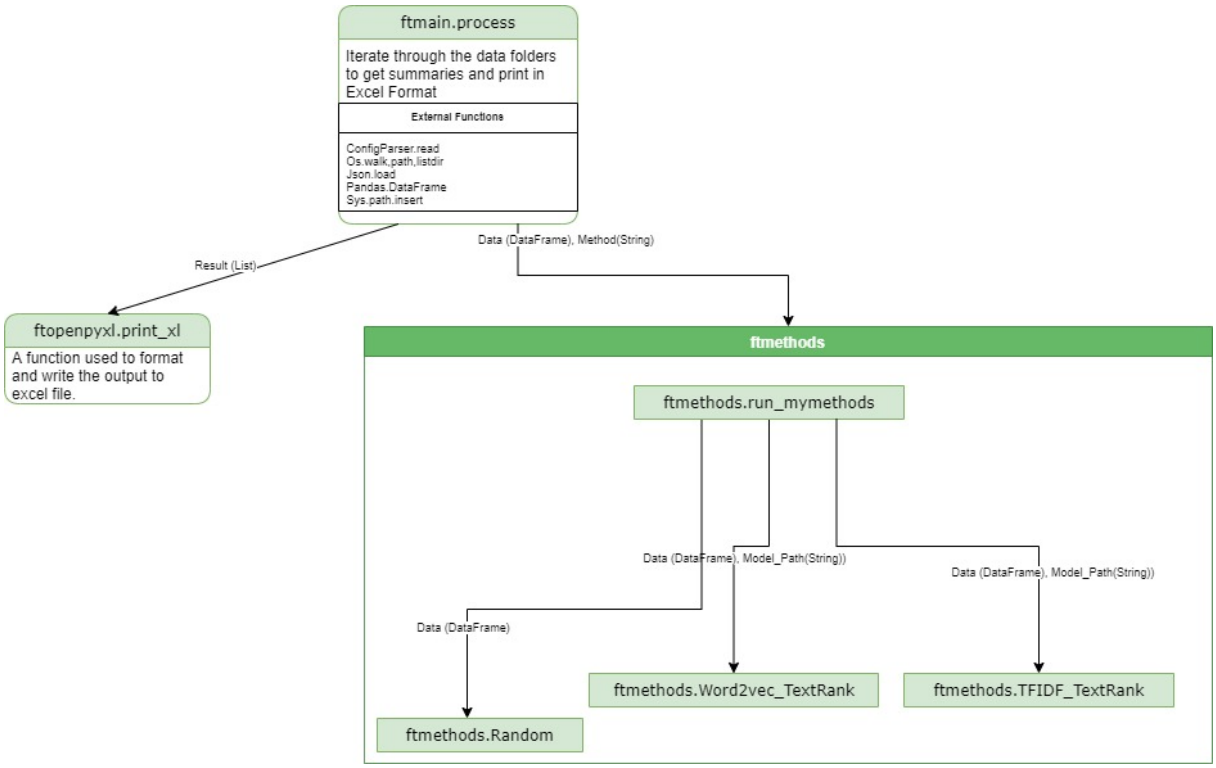

Figure 13. Overview of generating and saving summaries.

Let us now see the functionality of each file in detail.

ftmain.py has one function that interacts with six internal functions to produce the formatted output summaries. The file can be run as:

```
fartaj@ikitik:~/ftproject/main$ python3 ftmain.py
```

Figure 14. Running ftmain.py

Using the paths defined in config.ini file, the function iterates through each patient file in the physician folder and gets the episodes with the help of ftphysician.get_episodes function. For each episode, it iterates through each patient file in the nursing folder and gets the nursing notes using ftnursing.get_nursingnotes. If it finds more than 10 notes in an episode, it sends the list of notes to ftmethods.run_methods for producing summaries. The iteration stops when the defined episode limit is met and calls ftopenpyxl.print_xl function to format and save the output as excel.

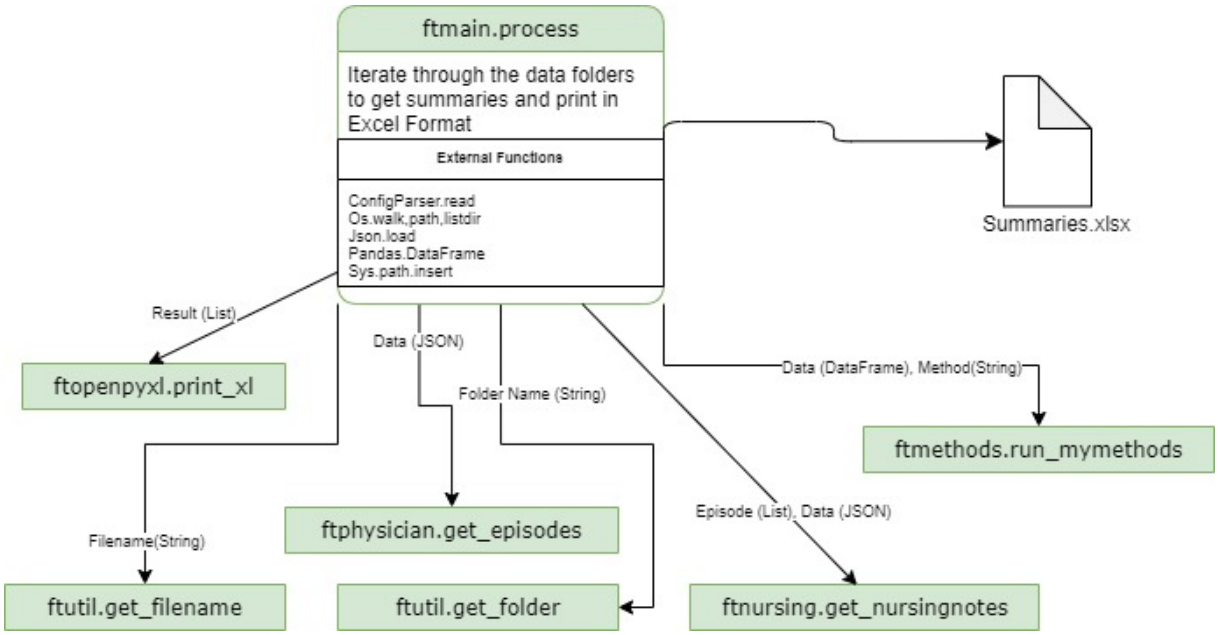

Figure 15. The main process - generating summaries

ftmethods.py has four functions. One for each model and an additional function managing the calls for the functions. From ftmain.process it receives the method name and nursing notes in a pandas data frame with columns “Original Notes”, “Reference Number” and “Meta data”.

The managing function namely run_methods is responsible to check if the method name is enabled in the config file using ftproject/shared/config.ini. It then sets appropriate parameters and calls the functions corresponding to the method names.

For this experimental work, we have three models. ftmethods.Random method tokenizes the nursing notes using ftdatacleaning.tokenize and picks sentences by chance to produce the summary. ftmethods.Word2Vec_LexRank method and ftmethods.TFIDF_LexRank method work in a similar way with the difference in the featured model used for executing test runs. Both methods perform the following main steps:

1. Tokenize the original nursing notes (containing meta data) and give each sentence a sequential number for reference.
2. Compile all the sentences in one string and prepare the data using ftdatacleaning.prepare_data function.
3. Using the model path (Word2Vec or TFIDF), compute sentence vectors.
4. Derive the similarity matrix and form the graph.
5. Using the page ranking algorithm, compute the scores of each sentence.
6. Extract top 30 sentences mapping them to the original data containing the meta data. It uses the ftdatacleaning.clean_sentence function to help in mapping.

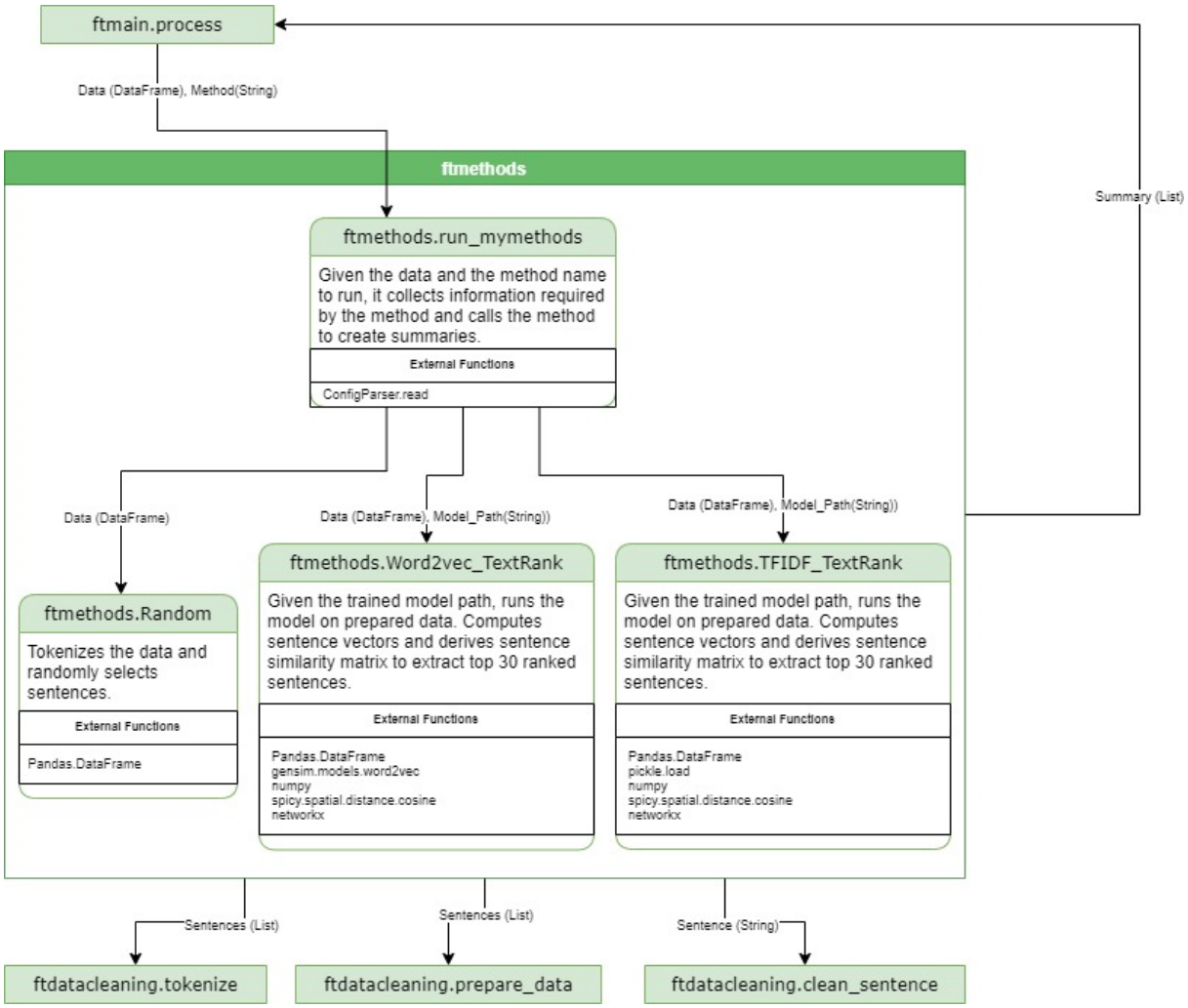

Figure 16. Details of Random, Word2Vec_LexRank and TFIDF_Lexrank algorithms

ftopenpyxl does not seem to a have a very complex structure or an important role, thus a diagram representing the code did not seem necessary to display. However, it is worth mentioning the structure of the excel file it produces and its sample display format.

The first sheet of the excel workbook contains hard-code text indicating the background of the study and guidance to the evaluation method. The sheets created after that are sequenced by episodes. Each excel sheet distinguished by its episode range and patient id, contains the original nursing notes and the summarized text of each method. The notes are extracted from four different kinds of tables which are indicated in the grey background headings before the original and summarized notes.

1.7 Shared

This folder (ftproject/shared) is meant to contain files that is common to more than one file of this project. The files in this folder are not meant to be used by other external projects. Apart from the config.ini file we discussed earlier, this folder also contains three other files namely ftdatacleaning, ftnursing and ftphysician. These are common python code files used both when creating the corpus and when executing the test runs by the trained models. ftphysician.get_episodes is responsible to extract the list of episodes given the physician JSON record. It uses ftutil.get_valuebykey function to iterate through the JSON record and returns a list containing multiple episode ranges (start date, end date).

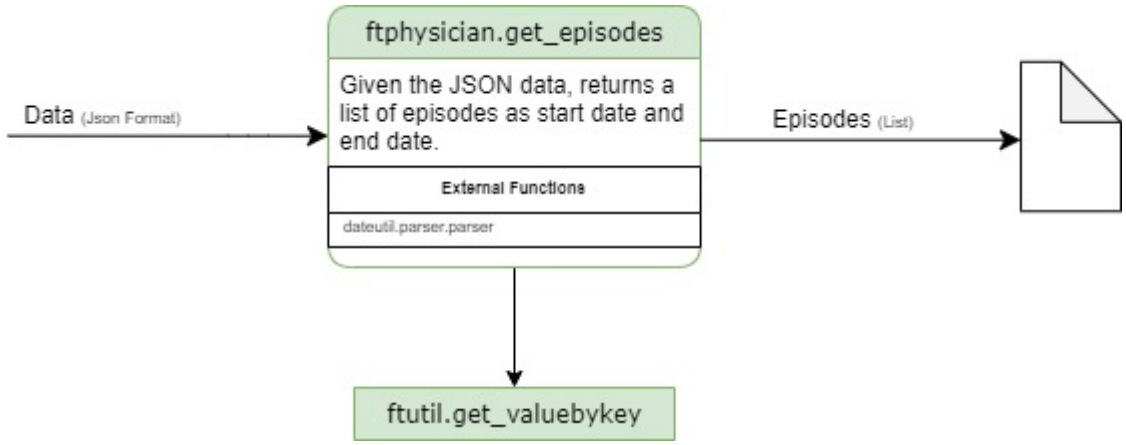

Figure 17. Fetching episodes from Physician Records.

Given an episode (start date, end date), and the nursing JSON record, ftnursing.get_nursingnotes is responsible to extract nursing notes and its meta data. It uses ftutil.get_valuebykey function to iterate through the JSON record and returns a list containing multiple notes and the meta data dictionary.

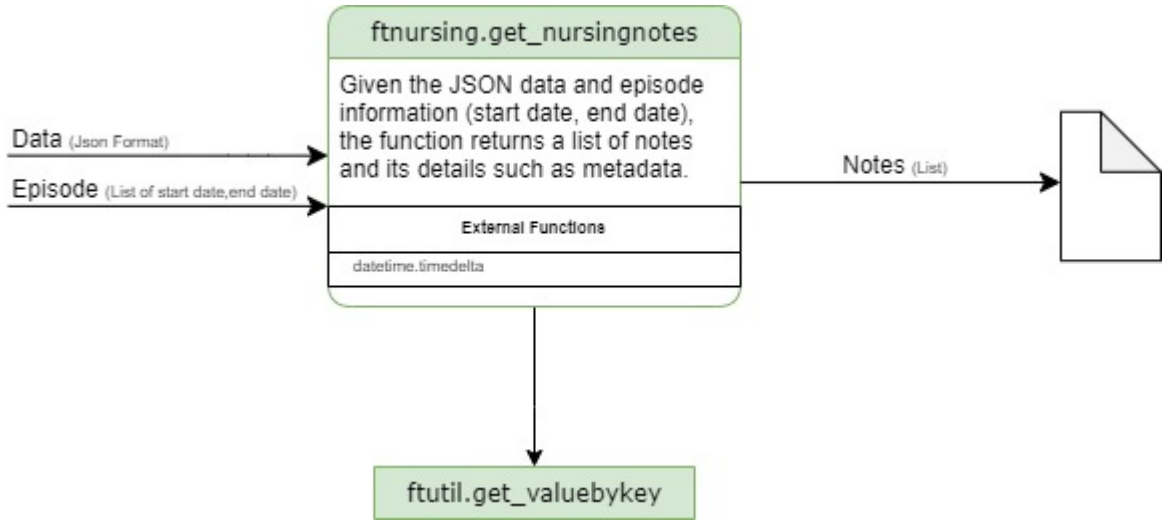

Figure 18. Gathering nursing notes based on Episodes

ftdatacleaning is responsible to help in tokenizing, removing punctuation marks and stop words. It caters these functionalities for text containing multiple sentences or single sentence. It consists of 5 functions each of which were made based on the necessity of its usage.

1. ftdatacleaning.tokenize uses the nltk.sent_tokenize function for finnish language and then splits on carriage return and newline using the regex expression "\\r\\n|\n" . Lastly it removes any empty sentences.
2. ftdatacleaning.clean_text removes any numbers and punctuations using the regular expression r'\d|[^\w\s]'. It also converts the text to lower case.
3. ftdatacleaning.remove_stopwords uses the nltk.corpus.stopwords function for finnish language to remove stop words. It also removes some very common found words such as 'teksti', 'otsikko', 'mg', 'iv', 'klo', 'mmhg', 'ml'.
4. ftdatacleaning.prepare_data is just a helper function that facilitates calling of tokenizing, then cleaning and then removing stop words.

The above 4 functions were used when multiple sentences were to be cleaned. For a single sentence, ftdatacleaning.clean_sentence was created which removed number and punctuations converting the text to lower case and then removed nltk.corpus.stopwords and the common words.

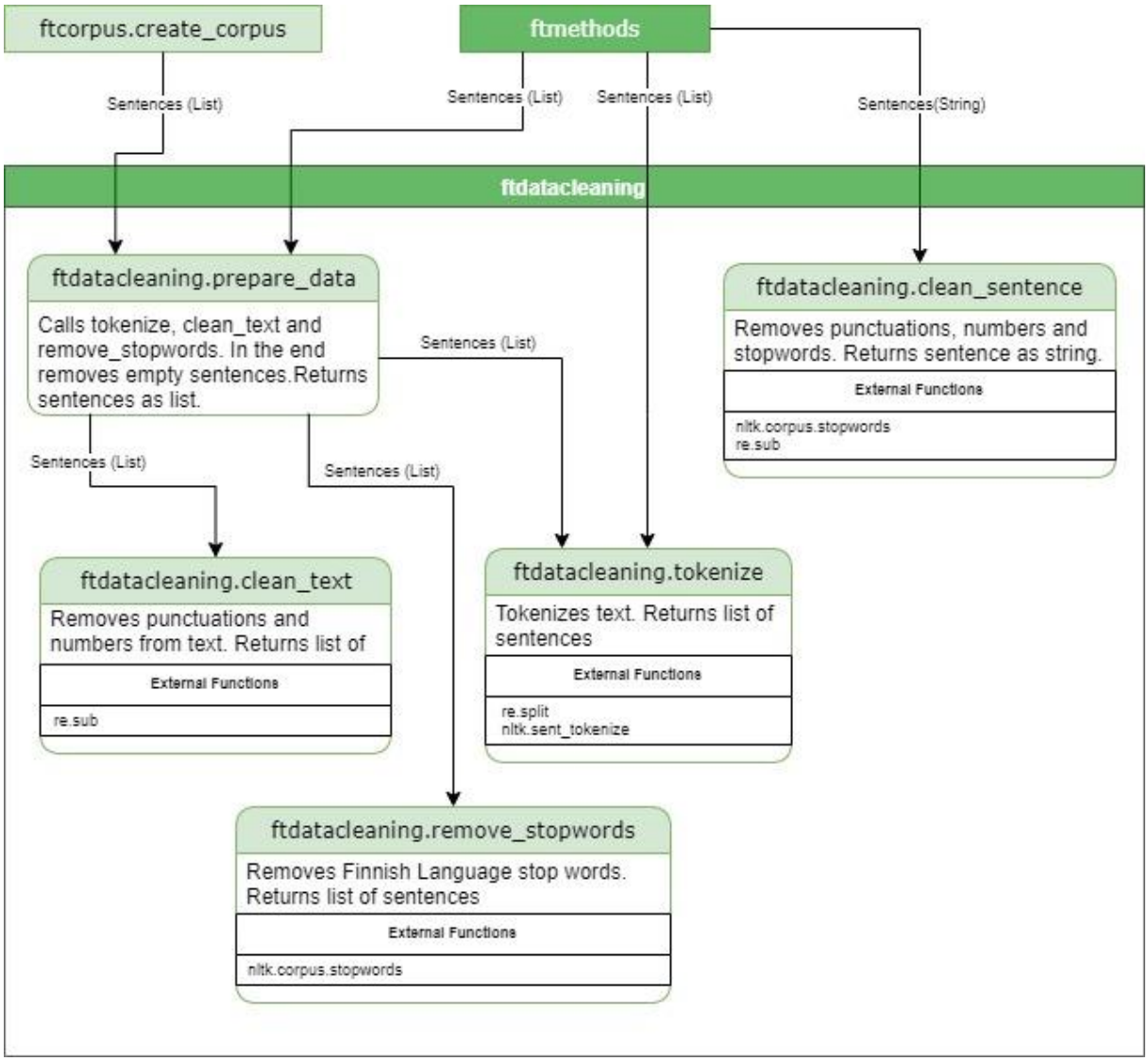

Figure 19. The shared Data Cleaning process

1.8 Utilities

This folder (ftproject/utilities) is meant to have simple functionalities that are shared among files in this project. The functions can be reused for other projects of the same dataset. It consists of only one file namely ftutil and the functions are responsible to help iterating through the JSON records (ftutil.get_valuebykey) or creating a text file given a list (ftutil.create_file). It also contains functions for getting the proper filename (ftutil.get_filename) or the correct folder name (ftutil.get_folder). All these functions are mainly called when creating a corpus or when running the test sets using the training models.

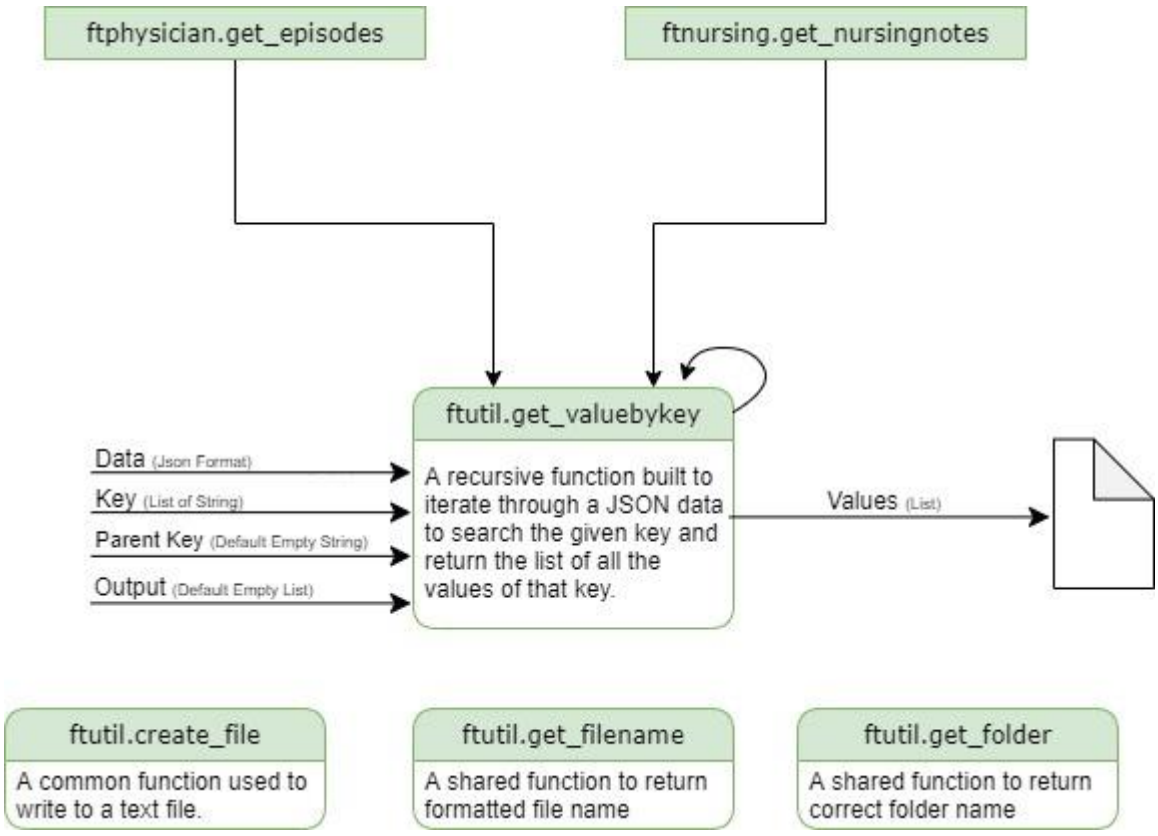

Figure 20. Common shared functions