University of Stavanger

FACULTY OF SCIENCE AND TECHNOLOGY

MASTER'S THESIS

Study programme/specialisation: Applied Data Science	Spring semester, 2021 Open
Author: Einar M. Langholm	
Supervisor(s): Martin Georg Skjæveland, Krisztian Balog	
Title of master's thesis: Constructing a Personal Knowledge Graph from Disparate Data Sources	
Credits: 30	
Keywords: Knowledge Graph, RDF, SPARQL	Number of pages: 85 Stavanger 15. june 2021

Abstract

This thesis revolves around the idea of a *Personal Knowledge Graph* as a uniform coherent structure of personal data collected from multiple disparate sources: A knowledge base consisting of entities such as persons, events, locations and companies interlinked with semantically meaningful relationships in a graph structure where the user is at its center. The *Personal Knowledge Graph* is intended to be a valuable resource for a digital personal assistant, expanding its capabilities to answer questions and perform tasks that require personal knowledge about the user.

We explored techniques within *Knowledge Representation*, *Knowledge Extraction/ Information Extraction* and *Information Management* for the purpose of constructing such a graph. We show the practical advantages of using *Knowledge Graphs* for personal information management, utilizing the structure for extracting and inferring answers and for handling resources like documents, emails and calendar entries.

We have proposed a framework for aggregating user data and shown how existing ontologies can be used to model personal knowledge.

We have shown that a *Personal Knowledge Graph* based on the user's personal resources is a viable concept, however we were not able to enrich our personal knowledge graph with knowledge extracted from unstructured private sources. This was mainly due to sparsity of relevant information, the informal nature and the lack of context in personal correspondence.

Acknowledgements

I would like to express my gratitude to my supervisors Martin Georg Skjæveland and Krisztian Balog for all their help and guidance throughout this project.

I would also like to thank my family for their support and patience.

Contents

Abbreviations

Acronym	**W**hat (it) **S**tands **F**or
KG	**K**nowledge **G**raph
KB	**K**nowledge **B**ase
KBP	**K**nowledge **B**ase **P**opulation
IE	**I**nformation **E**xtraction
RE	**R**elation **E**xtraction
PKG	**P**ersonal **K**nowledge **G**raph
NER	**N**amed **E**ntity **R**ecognition
EL	**E**ntity **L**inking
PIMS	**P**ersonal **I**nformation **M**anegement **S**ystem
DPA	**D**igital **P**ersonal **A**ssistant
RDF	**R**esource **D**escription **F**ramework
RDFS	**R**esource **D**escription **F**ramework **S**chema
OWL	**W**eb **O**ntology **L**anguage
POS	**P**art **O**f **S**peech
NLP	**N**atural **L**anguage **P**rocessing

Chapter 1

Introduction

1.1 Motivation

Through our digital presence our interests, communication, activities and opinions are better documented than ever before. A large part of this data is, in theory, easily accessible from any digital device. However, this data is for most of us not contained withing one singular source, but fragmented across a large number of services like email providers, social networks, streaming services and cloud services. The result is that personal information can be hard to manage and utilize.

To address this problem, *Personal Information Management Systems* aim to make all the users information accessible from one source, connecting data from multiple providers to the users personal server and making it available from one interface [1]. These systems focus on organizing data and typical do not model the information contained within this data to be machine readable, which modern interfaces like digital personal assistants would benefit greatly from.

Digital personal assistants have gone from entertaining novelties to useful tools in just a few years. Systems like Google Assistant[1], Microsoft Cortana[2] and Apple Siri[3] are able to answer practical questions, entertain and perform specific tasks. The systems can be given access to emails and calendars and are subsequently able to answer questions regarding appointments and to read out emails. Asking these systems questions like "Who are George W Bush's siblings?" shows how well these systems can interpret natural language queries and how they can utilize well-structured knowledge graphs to answer. Paradoxically, personal assistants have limited information about its own user that can

[1]https://assistant.google.com/
[2]https://www.microsoft.com/en-us/cortana
[3]https://www.apple.com/siri/

be utilized for answering queries and perform tasks. This is because only a small fraction of the user's data is well structured with meaningfully defined relationships. This, along with the fact that digital assistants typically only have access to a fraction of its user's information, leaves room for possible improvements for these types of systems.

1.2 A Personal Knowledge Graph

The motivation behind this project is the idea of a *Personal Knowledge Graph* as presented by Balog and Kenter [2]: A user-centric graph consisting solely of information about entities of interest to the user and how they relate to the user, i.e., entities such as persons, events, locations and companies interlinked with semantically meaningful relationships in a graph-structure where the user is at its center. The graph is created by extracting information from structured, semi-structured and unstructured sources using techniques from natural language processing. A digital personal assistant (DPA) can then utilize such a graph to answer questions and perform tasks that require reasoning over personal information.

Queries

- What is my mother's address?
- What bus can I take to mom?
- Where can I get a new drive belt for my scooter?
- Where did I buy my blender?
- Are there any interesting concerts next week?
- What is the number to my plumber?

Tasks

- Call my plumber.
- Send the images from my mothers birthday to my father.

1.3 Objectives

This project aims to conceptualize a framework for aggregating user data into one uniform source of personal information. We want to explore methods for information extraction on private data and model this knowledge in a graph centred around the user, containing only entities and relationship of personal interest.

Is a Personal Knowledge Graph constructed from the user's personal sources, a viable concept? Which challenges need to be overcome regarding information extraction and knowledge representation?

To reach our goal the following key challenges need to be addressed:

1.3.1 Ontology

We need to define the concepts found in our Personal Knowledge Graph, depicting the properties, relationships and logic necessary to describe our domain. An ontology capable of modelling personal details like opinions and preferences, family and personal relationship intertwined with users recourse like emails, document and correspondence. We need to find out if any existing ontologies can be used for this purpose or if it is necessary to create a new ontology from scratch or expanding and merging existing ontologies.

We also need to find a way to evaluate if it is free of inconsistencies and capable of capturing the desired knowledge.

1.3.2 Building the knowledge graph

In order to populate our graph we need to obtain personal data, extract structured information from these sources, map the structured data to our ontology and finally populate our PKG.

1.3.3 Using the Personal Knowledge Graph

We want to know how the personal knowledge graph can be used, its advantages and disadvantages over comparable solutions and if the information extracted from our sources is sufficient to create a knowledge source large enough to be of any practical use.

1.3.4 Conceptualising a framework

We need a framework capable of consciously obtaining/synchronizing data from multiple disparate sources, running the appropriate processing on the data and populate the PKG.

The framework needs to be accessible for the user to query, update and maintain their PKG.

1.4 Contributions

1.4.1 Ontology

We have created a knowledge structure for personal information that is centred around the user and tailored to answer queries that is personal in nature, rather than general. The relationships defined in the PKG are fine grained, including detailed family and personal relationships. It also includes personal abstract concepts like opinion. The PKG ontology is a combination of well known general purpose and domain specific ontologies expanded for the purpose.

1.4.2 Building a Personal Knowledge Graph

Information Extraction We have evaluated which personal resources contain useful information in the context of a *Personal Knowledge Graph*, how and if this information can be extracted and utilized. This includes information from structured, semi-structured and unstructured sources. We assessed techniques within information extraction, testing pre-trained NPL-pipelines and looked into state-of-the-art methods created specifically for informal text.

Our research into Information Extraction on informal text, revealed that the amount of information relevant for the PKG contained within personal correspondence may be sparse. The lack of context and informal nature of the text make the usable knowledge possible to extract using IE techniques minimal.

Entity Linking We have shown how a gazetteer enriched with informal and formal identifiers for entities contained within the personal knowledge graph can be used in the EL-pipeline for simplifying mention detection in personal text and minimizing problem with ambiguity under disambiguation.

Data enrichment .

We used domain specific sources for data enrichment and Entity Disambiguation. Expanding our graph by identifying and extracting additional information from domain specific sources. Linking entities of interest extracted from a service provider to global identifiers and resources on the web. This connects entities in the personal knowledge graph to general knowledge graphs, expanding its usability.

1.5 Thesis structure

Chapter 2. Preliminaries gives a brief introduction to the basics of knowledge representation, techniques and standards in addition to a short introduction to *Information Extraction.*

Chapter 3. Related Work contains an overview of relevant research related to personal Information Management systems, personal knowledge graphs and knowledge graph population.

Chapter 4. Solution Approach Details our approach towards developing an ontology, how we built our own personal knowledge graph from private data, how we created our framework and how we use the PKG.

Chapter 5. Experiments and Results In this chapter we look in to information extraction on personal correspondence using public datasets. We evaluate our proposed knowledge structure, our approach towards populating a personal knowledge graph using our private data and the resulting personal knowledge graph's practical use.

Chapter 6. Discussion In this chapter we will discuss our findings from our work towards building a Personal Knowledge Graph.

Chapter 7. Conclusion and Future Directions

Chapter 2

Preliminaries

This chapter contains a brief introduction to the basics of knowledge representation. Techniques and standards are included to give a foundation on topic prudent to this theses, but not a part of the curriculum in the Applied Data Science masters program at UiS. We also give an overview of the components of *Information Extraction*

2.1 Knowledge Representation

For machines to utilize information for answering questions and performing tasks, it is crucial that the information is structured in a way that makes sense for machines. *Knowledge representation* refers to the task of rendering human knowledge and reasoning into a language that enables it to be processed by information systems [3]. There are four main techniques that are used for this task, *Logical representation*,*Production Rules*, *Frame Representation* and *Semantic Network representation.*

Semantic Network Representation is the Knowledge representation method that the techniques and standards utilized in this project has evolved from. These are graph like structures, where the nodes represent concepts or distinct entities and the linkage between them are their semantic relationships. Concepts are entity types/classes that group individuals with similar characteristics together, like Person, Organization, Pet, Cat or Dog. The properties and semantic relationship that interlink them hold true for all instances of the class.

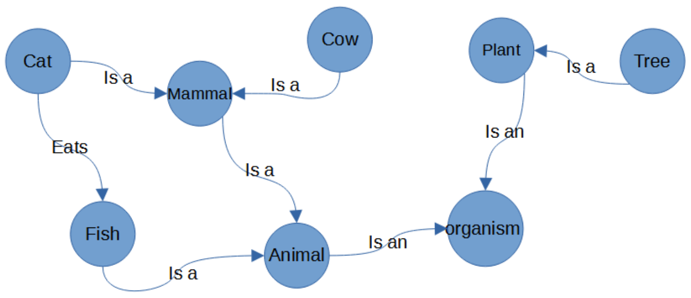


Figure 2.1: A toy example of semantic network describing the relationship between a few concepts.

Entities can be representations of *Named Entities* like a specific person, event, group or organization, or they can represent abstracts entities like emotion, wind or distance [4].

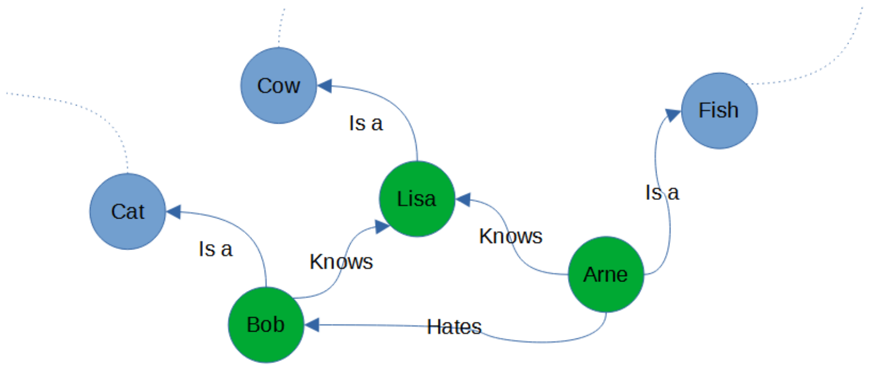


Figure 2.2: A toy example of semantic network describing the relationship between a few instances. Lisa, Bob and Are are instances of the classes/concepts Cow, Cat and Fish respectively. Along with Figure 2.1,

2.2 Ontology

In the context of *Information Science* ontology is the formal description of concepts within a domain, their properties and the relationships between them. Hierarchical structures allows for inheritance from high level conceptual classes to low level classes and their instances. Defined relationships, reasoning rules and restriction ensures data coherence and allows for inference. They can be formalized using any Knowledge representation language that has the desired expressivity, rigour and semantics [5]. Languages based on Description logics (DL) is the most common.

$$
\begin{aligned}
Cat &\sqsubseteq Mammal &&\longrightarrow \quad \text{Concept inclusion, All Cats are Mammals} \\
Cow &\sqsubseteq Mammal && \\
Mammal &\sqsubseteq Animal && \\
Cat &\sqsubseteq \neg Cow &&\longrightarrow \quad \neg = \text{Negation, Cats cannot be Cows} \\
Cat &\sqsubseteq \exists eats.Fish &&
\end{aligned} \tag{2.1}
$$

The set of logical descriptors in 2.1 is formulated in the description logic $\mathcal{ALC}$ (Attribute Language with general Complement) .

An ontology often models the conceptualisation of a domain only, however they can include unique entities, which are the individual instances of the classes [6]. They may refer to a specific person, location, organization and so on.

> An ontology defines (specifies) the concepts, relationships, and other distinctions that are relevant for modelling a domain. The specification takes the form of the definitions of representational vocabulary (classes, relations, and so forth), which provide meanings for the vocabulary and formal constraints on its coherent use.
>
> *Tom Gruber 2009 [7]*

Ontology authoring Designing and formalizing an ontology is known as ontology authoring. Ontologies aim to capture the knowledge within a domain and requires a deep understanding of the domain, in addition to expertise within logical modelling. This is a non-trivial task with no singular solution for modelling a domain.

2.3 Knowledge Graphs

The term *Knowledge Graph* KG was introduced as the name of a system developed in the 1980s using closed domain *Semantic Networks.* i.e The relationships in the KG was restricted to a predefined set. This facilitates logic-based knowledge representation and the use of algebra on the KG [8]. To this day *Knowledge Graph* does not have a clear cut and accepted definition, and given the growing popularity of these graphs there has been some research towards defining them like Ehrlinger and Wöß [9].

The term is typically used to describe graph structured knowledge representations where both the conceptual ontology and a set of instances are represented. There is a large number of systems built for different purposes using different technologies and architectures that are described as KGs. However this term is commonly associated with graphs built using the standards developed for the *Semantic web*.

2.3.1 Semantic web technologies and Linked Data

Semantic web is the term used by Tim Berners-Lee to describe the idea of a machine readable world wide web (Web). It is an extension of the web, formatting the data in a way so that machine readable knowledge can be shared cross the internet. The semantic web can be seen as a collection of interlinked knowledge graphs, rather than the collection of interlinked documents and resources the traditional web consists of. This is achieved by a common standard for *Knowledge Representation* using *Knowledge Graphs*.

The *Semantic web technology's* refers to the technology stack and standards defined by World Wide Web Consortium[1] (W3C) for supporting the *Semantic web*. Data structured and published following these standards are referred to as *Linked Data*.

RDF

The *Resource Description Framework* (RDF)[10] is the *w3c* standard depicting the base structure for the *Semantic web*. The specification details a framework for how to represent resources and the relationship between them, building on the URI standard. In the RDF-framework each relationship is represented by a triple consisting of a subject, predicate and a object. The object is a unique identifier for a resource as a URI or a blank node. The predicate describes the relationship between the two resources and are represented by a URI. The object can refer to unique resources via an URI or a blank node, or it could contain the information itself in the form of a literal, a text string, a number or a date. This simple framework allows for merging of data when the sources has different structures.

Subject	Predicate	Object
<Bob>	rdf:type	<Cat>
<Bob>	Knows	<Lisa>

[1]https://www.w3.org/

RDFS & OWL

RDF Schema RDFS is an extension of the RDF standard, providing a way of formally describing the fundamental building-blocs of an ontology. Concepts denoted as classes can be defined and described using this framework. This includes describing class properties like class hierarchies, the relationship between them and their utility properties.

The *Ontology Web Language* OWL is the framework developed to facilitate logical reasoning, expressing knowledge in the the *Semantic web*. This knowledge representation language(s) extends RDFS, by adding reasoning rules and restrictions necessary to describe an ontology. This is the framework that facilitates for machines to do complex inference on the data in *RDF* graphs.

Subject	Predicate	Object
<Cat>	rdf:Type	owl:Class
<Cat>	rdfs:SubClassOf	<Mammal>
<Cat>	owl:disjointWith	<Cow>

Named Graphs

A *Named Graph* [11] is a collection of RDF triples identifiable by an URI. The identifier is used to anchor the collection of triples to metadata giving it context and provenance. This allows us to distinguish triples from different sources and keep track of versions with timestamps.

RDF-STAR & SPARQL-STAR

RDF-star[12] proposed by Hartig is a small extension of RDF, allowing for a RDF triplet to be included as a subject or a object in another triplet. This makes it possible to annotate a triplet with temporal information, weighted scores, source and other relevant contextual information directly. This extension is relatively new and is not formalised as a W3C standard. There is established methods to way to add this type of information in RDF is to use named graphs and rectification. These methods are more involved than RDF-STAR, and thus makes the queries more complicated.

Subject	Predicate	Object
< <Subject> <Predicate> <Object> >	Predicate	Object
< <Bob> <knows> <Lisa> >	said	<Arne>
< <Bob> <knows> <Lisa> >	since	<Thursday>

2.3.2 Public knowledge graphs

DBpedia[2] and YAGO[3] (Yet Another Great Ontology) are examples of public general purpose knowledge bases built using *Semantic Web Technologies.* These are cross domain encyclopedic knowledge graphs containing structured information extracted from Wikipedia and other resources like WordNet and GeoNames. The data is published under open data licensing and can be used to answer specific questions direct, or they can expand the reach of a purpose built knowledge graph.

2.3.3 Populating knowledge graphs

A Knowledge Graph is built by populating an ontology describing the properties and relationships between concepts/classes with instances of these. This require the input data to be structured, for the equivalent classes of entities to be known and relationships and properties of these entities to be known.

If the source-data is structured, this task requires mapping the source schema/ontology to the target ontology. If the data is unstructured, text the task require identifying the relevant entities in the text along with their properties and relationships. This can be done manually or it can be automated using NLP-techniques, a task called *Knowledge Base Population*

2.4 Knowledge Base Population

Knowledge Base Population KBP is the task of enriching or building a knowledge graph by extracting information from unstructured or semi-structured sources [4]. Most contributions to this field are focused on extracting and structuring information from large text corpora like online encyclopedias and collections of news articles. TAC[4] had an annual KBP Trac from 2010 to 2017, which has resulted in significant improvements

[2]https://www.dbpedia.org/
[3]https://yago-knowledge.org/
[4]NIST TAC Knowledge Base Population https://tac.nist.gov/2017/KBP/index.html

within this field. This is still a challenging task, with state-of-the-art methods performing well below humans, even on formal texts that are generally well written and rich in facts Mesquita et al. [14].

KBP involves multiple sub-tasks that can be divided in to two main components; *Entity Linking* and *Relation Extraction.*

2.4.1 Entity Linking

Entity linking is the task of uniquely identifying entities in a body of text. [15]. The task consist of detecting mentions of relevant entities in text and then disambiguate these extracted entities to the corresponding entities in a knowledge base. Entity Linking is commonly divided in to three sequential tasks; Mention detection, Candidate selection and Disambiguation.

There are multiple public datasets created for training and evaluating entity linking systems. The common approach is to use public encyclopedic articles as the text source and link the mentions in these texts to the corresponding article describing them or to the entity representation in a public knowledge graph like YAGO or DBpedia.

Mention Detection & Named Entity Recognition

Mention Detection is the task of finding words or a sequence of words in a text that may refer to an entity, this is typically the first step of *Entity Linking. Named Entity Recognition* NER identify entity mentions in text and labels the mentions with a class. This narrows the range of the mention detection to a predefined set of entity types like Person, organization and Location. This reduces the task of uniquely identifying these mentions, disambiguation.

As for EL, most research on NER has been focused on formally written text from encyclopedias and news articles. There are however a few papers focusing on the challenges with informally written text.

Ritter et al. [16] focuses on the challenges with NER on short informal text, tweets. The paper outlines a NLP-pipeline created specifically for this type of data. The model shows significant improvement over out of the box solutions which all preform poorly on informal text. NER on tweets is also the subject of Etter et al. [17]. In addition to the challenges related to short informal text, this model also aims to be useful on multilingual data, using only language independent features to identify 10 entity types.

The Enron corpus is one of the few existing publicly available datasets containing genuine personal communication. It has therefore been used to train models within several NLP-tasks requiring informal communication. Most relevant are the studies involving KBP and NER. Minkov et al. [18] uses this dataset to train a model for NER, the model is specifically created for NER on emails, leveraging the structure of emails to improve the result.

Co-reference Resolution

Identifying all mentions referencing the same entity in a text requires finding all expressions that refer to this entity. Expressions might be name-variants or pronouns like I, me, you, it, him and her. This is called co-reference Resolution.

Candidate selection

Collecting a set of possible entities that may correspond to a entity mention in the text is a task called *Candidate selection* [4].

Entity disambiguation & Named Entity disambiguation

Entity Disambiguation refers to the task of uniquely identifying a entity mention from a set of possible entities. The best candidate may be selected using similarity measures using the context given in the text, entity-relatedness using other entities detected in the text or context independent features relying on statistics of the mention and the entity alone. *Named Entity disambiguation* uniquely identifies entity mentions that's already classified, i.e processed by a NER-model.

2.4.2 Relation Extraction

Relation Extraction is the task of detecting semantic relationship between entities like persons, organisations and locations from mentions in unstructured text. Relation extraction is commonly reduced to a closed domain task by defining the set of relationships and properties to be extracted. The task can then be viewed as a slot-filling, where the challenge is to classify the predicate occurring between entities in the text. Due to error propagation the end-to-end performance of pipeline models is quite low, even if performance of each individual sub-task is close to human performance [19].

Chapter 3

Related Work

This chapter contains an overview of relevant research related to personal Information Management systems, personal knowledge graphs and knowledge graph population.

3.1 Personal Information Management

Research in the field of *Personal Information Management Systems* PIMS and *Personal Data Space Management Systems* PDSMS, focus on giving users better control over their data, both regarding information security and in the practical aspect of managing personal information [20]. These systems aim to provide one interface for multiple disparate data sources, allowing the user to synchronize, update and perform global searches across all their data from one application.

NEPOMUK [1] Groza et al. [21] and ISISIR [2] Cheyer et al. [22] describe variants of *Semantic Desktop Systems*, these are essentially PIMS that utilize semantic web technologies as a backbone structure. These ontologies are centered around the user and modelled to interlink user's resources like emails, documents and calendar entries with meaningful relationships.

The *Solid project*[3], is an effort to create a specification for managing and sharing personal data using graphs. The idea is to give the user full control over their own data, letting them administer access to any slice of their information. The specification would negate the need for complicated data mapping and transformations, as both users and providers would be using the same base vocab, standard data formats and protocols. Like PIMS, this project is also focused on the user's resources.

[1]NEPOMUK Networked Environment for Personal, Ontology-based Management of Unified Knowledge
[2]ISISIR Integrate. Relate. Infer. Share
[3]https://solidproject.org/

There are also a growing number of commercially available PIMS, however integration from external sources are limited to a handful of predefined applications and the user is made dependent on the selected provider.

The primary focus of PIMS are the user's resources, not necessarily representing entities and relationships relevant for the user that exist within this data.

3.2 Digital Personal Assistants

Digital Personal Assistants are software agents designed to help the user by answering questions and preforming tasks. Modern DPAs are able to interpret questions and commands from human speech and respond verbally. They can place phone calls, read emails, give directions and answer questions about weather forecast, historical events and other general knowledge. They are also capable of modeling basic family relationships and conduct logical reasoning using this.

DPAs can be integrated as software running on electronic devises like Google Assistant[4] found on most android devices, Apple Siri[5] on iOS devices and Cortana[6] for Windows PCs. Amazon Echo, Google Nest and Apple HomePod are stand-alone devices where DPA-functionality is primary.

According to Microsoft Voice report 2019 [23], 72% of respondents reported using DPAs, however the survey also revealed that 40% of the users reported concerns around trust and that consumers are hesitant to share personal identifiable information to their DPA.

Luger and Sellen [24] aims to understand what limits and motivates the use of DPAs, based on interview with 14 regular users. Their findings shows a discrepancy between user expectation and system operation. The DPAs where mostly used for basic tasks and the users limited understanding of the PDAs knowledge and capabilities hindered them from fully utilizing their potential.

3.3 Personal Knowledge Graph

The idea of extracting personal information from a user's devices and services, subsequently aggregating the data into a knowledge base of personal information has been the subject of several recent research papers. This thesis was inspired by a paper named

[4]https://assistant.google.com/
[5]https://www.apple.com/siri/
[6]https://www.microsoft.com/en-us/cortana

’Personal Knowledge Graphs: A Research Agenda’ by Balog and Kenter [2]. It defines the concept of a Personal Knowledge Graph and formulates a few key research questions. The position paper describes some of the challenges with extraction, data aggregation and structuring of personal knowledge, in addition to giving a clear motivation behind a Personal knowledge graph. However it does not propose any solutions for the challenges.

> We define a personal knowledge graph to be a source of structured knowledge about entities and the relation between them, where the entities and the relations between them are of personal, rather than general, importance. The graph has a particular “spiderweb” layout, where every node in the graph is connected to one central node: the user.
>
> *Balog and Kenter Personal Knowledge Graphs: A Research Agenda*
> *Krisztian Balog, Tom Kenter*

Research towards populating a PKG as defined by Balog and Kenter has been done by Vannur et al. [19], Tigunova [25] and Tigunova et al. [26] among others.

3.4 Knowledge representation for Personal knowledge Graphs

To our knowledge there are no ontology for personal information that confer with the definition of a Personal Knowledge Graph used in this thesis. Montoya et al. [27] includes the description of an ontology used in their personal management system. The ontology built by extending the scherma.org vocabulary is indeed an ontology for personal knowledge, however the ontology is built from the perspective of personal management system, entities are closely related to the source and the ontology allows for duplicate entities.

The friend of a friend (FOAF) ontology is a well known and widely utilized ontology for describing relationships between persons, agents and object [28]. The ontology is built around the FOAF profile, which is a set of properties describing an entity such as a person or agent. This ontology contains personal information and relationships of personal interest, but its created to describe a social network, it is not centered around one user and it is not wide enough for the proposed personal knowledge graph.

Kargioti et al. [29] describes an ontology focused on personal knowledge management centered around extending the Person entity from FOAF. They call their ontology Onto-Life and its purpose is to model a person’s life by describing the person’s characteristics, relationships and experiences(sit).

The patent *Structured user graph to support querying and predictions* filed by Google, sketch the use of a knowledge graph that is focused on information about the user. They depict a knowledge graph that contain the user's personal interests, likes and dislikes. The focus of this is to improve the services they provide, namely returning more relevant search results and advertising. There is however one major caveat about *'Personal' Knowledge Graphs* generated by these type of service providers, they are entirely created on the premise of the provider, tailored to the services they provide and limited to the data within their domain.

3.5 Knowledge base population for Personal knowledge Graphs

Prior work within KBP focusing on extracting personal information includes extraction from personal communications like emails [27, 30] and chat [25, 31, 32], conversations with personal assistants[33, 34], push notifications [35] and structured information like location data [27]. Naturally, access to large datasets containing personal conversations for model-training and evaluation is very limited. Research focused on extraction from dialog has therefore used public sources like topic specific public chats [32, 36, 37], interview transcripts [38] and blog-posts [39] in addition to alternatives to genuine conversations like movie scripts [36] and query logs and retrieved snippets [40].

Informal text

One of the challenges with extracting personal information from dialog is that informal text like personal conversations contain far less explicitly stated facts compared to formal text. Tigunova et al. [32, 36, 37] proposes models to infer participants personal attributes from dialog. (*Author profiling*) Their models utilize latent textual ques to predict two relationship-types, the subjects profession and hobby. One of the models called CHARM includes an external document collection in the pipeline to be able to predict hobbies and profession not seen in the training-data. In short, the users utterances are scored in terms of relevance and the top ranking terms are used to query the external document collection. The top ranked documents are then used to predict the attribute value. It is assumed that the documents in the collection can be automatically associated with relevant attribute values. The method shows tangible results, but the authors do acknowledge that the dataset used as proxies for conversational data are somewhat biased. Selected Reddit forums used in [32, 36, 37] and TV and Move scripts used in [36]. As external document collection they use Wikipedia and the web. A live demonstration[7] of the CHARM model

[7] https://d5demos.mpi-inf.mpg.de/charm

[32] is made available for two scenarios, interaction with a chat-bot and on user's posts on social media [26].

Li et al. [40] proposes a pipeline-model for personal knowledge graph population from user utterances. The first step is a personal assertion classifier(SVM-model), this model predicts if a given statement contains relevant personal information. The second step is relation detection, assigning the assertion in to one of the relationship classes(SVM-model). The final step is slot filling (CRFs). They create a purpose-build dataset using Freebase triples to mine for snippets containing the information in Natural language form using a search engine in addition to annotated utterances from Cortana query logs.

Digital assistants are frequently mentioned to be the key reason for generating a personal Knowledge graph. Popular digital assistants are capable of extracting personal information from interactions with their user. For instance, Google assistant will store the relation you have to a contact if you explicitly state it to the assistant. ("Marta is my mother"). Agarwal et al. [33] proposes a model that can reason over previous interaction with a digital assistant, focusing on queries containing businesses and locations. The method stores entities and relevant metadata retrieved by a query in a graph, the information can then be utilized in later queries. Research into personalizing chit-chat chat-bots has also focused on making the interaction between the user and the bot more personal by extracting and using information about the user in the conversations. Here the bot actively engages with the user asking relevant questions.

Formal text

The limited amount of research within KBP from conversational text and the lack of a public purpose-built dataset for evaluation makes it impossible to directly compare existing methods based on published research alone. Vannur et al. [19] use the TACRED [8] dataset, a public set created for evaluating general KBP systems, built from newswire and web text. Their goal is to create a model for populating a Personal Knowledge Graph and they propose methods for reducing bias in the training data.

Semi-structured data

Li et al. [35] explores the possibility of extracting personal information from smartphone push notifications. Push notifications are generated by multiple sources and contains short summaries of information relevant for the user. This method negates the difficulties

[8]https://nlp.stanford.edu/projects/tacred/

with aggregating data from multiple sources with different data-structures. Their system learns the patterns of notification templates, identifies template parameters, predicts the semantics of the template and uses this to define mappings from notification templates to SPO triples. Their model extracts 11 types of relationships within 4 categories; user profile, social, location and shopping.

In addition to the research focusing on knowledge-base population, there are some research focusing on creating a personal knowledge graph from the perspective of Personal Information management systems.

The paper "A knowledge base for personal information management" [27] and [41][42], details the framework of a complete system. The system aggregates data from multiple sources, performs ontology alignment and semantic enrichment and stores this information in a personal knowledge graph. The system uses data from emails, calendars and location history to extract relevant proprieties and relationships between three entity types; agents(persons,organisation), events and locations. Their solution called Thymeflow, is available as an open-source system. They leverage standards used for email[43] and calendars[44] to extract information in addition to Google Location data which has its own proprietary data structure. Their solution uses an ontology built on existing and established vocabulary; schema.org, which they have extended to allow for more detailed information about entity's relevant for the user. They have developed purpose-built techniques for ontology alignment and entity linking. They consider the resulting graph as a view of the user's personal information, where the information extracted is closely related to the source, allowing for an entity being represented by multiple nodes in the graph.

In the paper 'Toward Activity Discovery in the Personal Web' by Safavi et al. [30] they use emails, documents, contacts and calendars as a source of information for generating a personal graph. This graph can also be seen as a view of the user's personal data. However, in their research they focus on extracting information (like noun-phrases) that will allow for Graph similarity search to extract entities relevant to a specific activity. Their goal is to develop methods that can be used for automating tasks like email organizing and prioritizing. For example by using the name of a project one would be able to extract emails, documents and contacts relevant to this specific project.

Emails

There are commercially available application that extract personal information from emails, however the focus of the knowledge extraction these applications is not to enhance the user experience. They focus on user data of commercial interest that can be sold.

The application can be offered to users for free, on the condition that they accept anonymized data to be sold. For example *Cleanfox* [45] helps you clean your mailbox, if you let the application extract information from confirmation orders, cancellation and Newsletters. *Edison* [46] is an email application that also extract and sells information from Newsletters and receipts, offering the user notification services, shipment tracking and travel alerts in return. What methods they use, how well they preform and how much detail is extracted from these type of service providers is not publicized.

3.5.1 Author profiling

The task of inferring/identify attributes about an author based on a collection of produced texts. Research into this field has focused on a few attributes like gender, age and occupation. Preoţiuc-Pietro et al. [47] aims to identify a Twitter user occupation based on their tweets. The high number of possible classes makes this a challenging task, even though this is just one relationship type.

Chapter 4

Solution Approach - My Personal Knowledge Graph

This chapter describes our approach towards building a PKG from personal data.

Conceptualising a framework The first section introduces an overview of the proposed concept, describes the tools and techniques used and describes the data used to develop and evaluate the PKG.

Developing the ontology With no known ontologies fitting to satisfy the requirement of our PKG, we found it necessary to develop a new ontology by expanding and merging excising ontologies. In section 4.2 we describe how the domain and requirements of our ontology was defined and how the resulting ontology is structured.

Building the Knowledge Graph Section 4.3 describes how we deal with the challenge of *Knowledge Extraction* on personal structured, semi-structured and unstructured data. We utilize available tools, techniques and standards to extract knowledge from personal sources and populate the PKG, creating one coherent source of information from multiple disparate sources. For each source type we describe what information we include, why it is included, how we obtain the data, how we process the data and how its modelled in the PKG.

Using the Knowledge Graph In section 4.4 we depict how we intend to use our PKG.

4.1 System overview

Inspired by the concepts by Montoya et al. [41] and the *Personal Information Management System* PIMS it describes, THYMEFLOW, our system is centered around a personal server connected to the user's data sources like email servers, personal cloud services and streaming services. PIMS are applications built to create one interface between the user, making it easier to manage resources like emails, documents, calendar entries and contacts cross platforms. Our proposed system aims to take this concept a small step further, creating a knowledge graph centered around the user, and all personal information of practical use. Enriching the graph with information extracted from these documents and information from additional services normally not considered for PIMS.

The *Personal Knowledge Graph* allows the user to define detailed relationships, personal opinions and views about entities of interest and intertwine it with resources like emails and documents. This facilitates document retrieval as well as allowing for direct questions to be answered by the graph.

Figure 4.1 shows an overview of the proposed system.

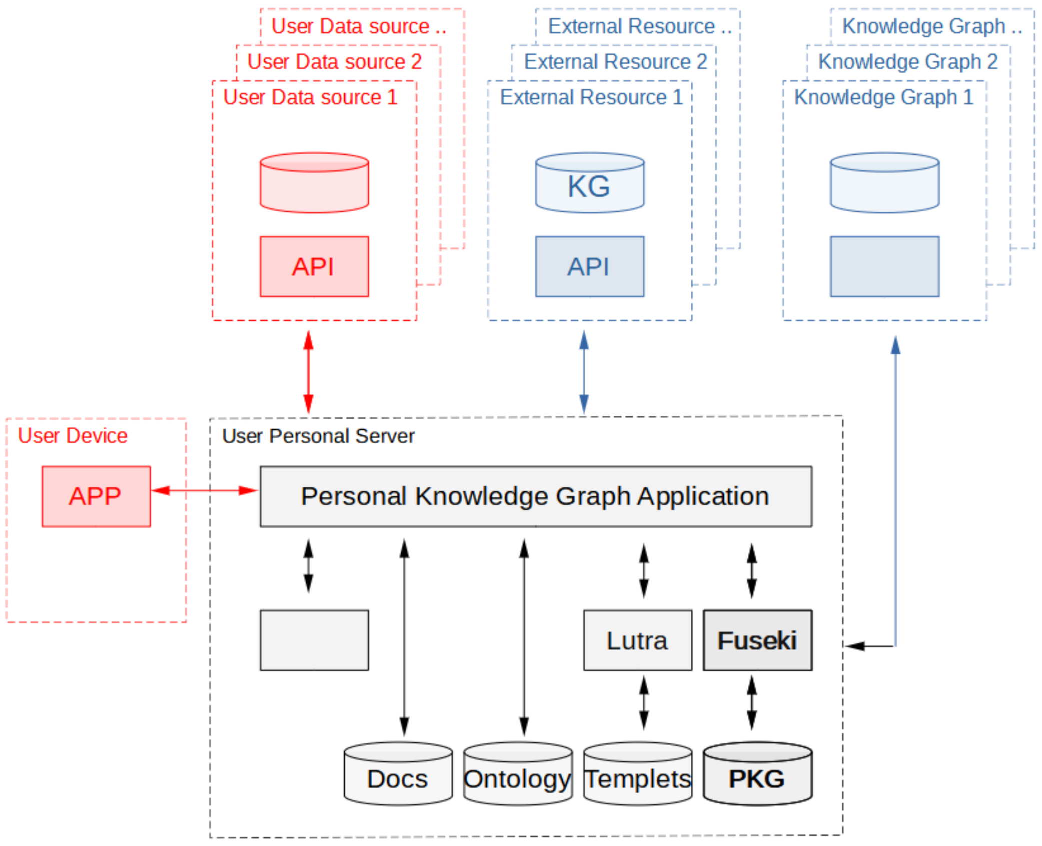


Figure 4.1: A simplified overview of the proposed framework.

Short explanation of the individual parts in fig 4.1

- **Personal Server:**
  - **Personal knowledge Graph Application:** The web-service that connects to the user's data sources, process the data and maintains the Personal Knowledge Graph.
  - **Fuseki:** Apache Jena[a] Fuseki is a SPARQL server used to update and query the Personal Knowledge Graph.
  - **Templates:** Templates are utilized to facilitate expanding the graph with new entities.
  - **Lutra:** is the tool used to read and write OTTR-templates [48].
  - **PKG:** The Personal Knowledge Graph
- **User Data Source:** Services containing the user's data. Email, Streaming Services, Social networks and Cloud services.
- **External resources:** External resources containing detailed structured knowledge within specific domains. Useful for Entity linking. e.g. MusicBrainz.org can be used to obtain global identifiers on artist in addition to linking the entity to other relevant resources on the web.
- **Knowledge Graph:** Public External knowledge bases like DBpedia, Wikidata and Yago. These are open domain structured encyclopedias.
- User Device: The device communicating with the PKG app.

[a]https://jena.apache.org/index.html

4.1.1 Tools and technologies

Knowledge representation

The knowledge representation method, technique's and standard used, where predetermined by the thesis proposal. The system is built around a *Knowledge Graph* following the standards of w3c for the *Semantic Web* [10].

Ontology authoring

The stand-alone application Protege [49] was used as tool for ontology development. WebVOWL [50] for visualization.

The Personal Knowledge Graph Application

The Personal Knowledge Graph Application was built as a basic web-service using Python and the Flask[1] framework. The application was set up for testing data collection, knowledge extraction and interactions with the Knowledge Base. The application in its current state is intended for testing only, with no usable interface developed. Jupiter Notebooks[2] with Python where extensively used .

Graph database

RDF-data is stored and accessed using so called triple-stores, these are essentially databases specifically created for RDF-triples. They are the applications that interpret the logical components defined by the ontology, making them capable of reasoning over the RDF-data, discovering implicit knowledge in the data. This project uses the open source Java framework Apache Jena and the SPARQL server Fuseki for this.

The graph database is configured with Jena *Full OWL Reasoner* [51].

Knowledge Extraction

Tools and publicly available solutions for *Knowledge Extraction / Information Extraction* and related tasks developed for Python where evaluated. For Python this

[1]https://flask.palletsprojects.com/en/1.1.x/
[2]https://jupyter.org/

include open-source tools and models from spaCy[3], Natural Language Toolkit (NLTK)[4], scikit–learn[5]and StanfordOpenIE[6]. In addition to this, GATE Developer[7] and GATEs Cloud services[8] where tested.

Templates

To facilitate transformation of instances described in table format to RDF graph structure, we used templates written as Reasonable Ontology Templates [OTTR][48]. *Lutra*[9], a tool for reading and writing these OTTR templates are used to transform instances described in tabOTTR[52] to RDF.

4.1.2 Dataset

Due to the personal nature of the data explored in this project, we do not have access to a public dataset that is suitable for building, training and evaluating our framework. We have therefore selected to divide the data into two.

Framework & knowledge structure dataset The first dataset is created for the purpose of developing and evaluating the *Personal Knowledge Graph*, this includes the methods utilized for collecting data, updating and querying the graph. The data is a collected from my own personal sources enriched with manually inserted data where necessary.

Information Extraction dataset The second dataset is a collection of public datasets included for the purpose of testing and evaluation NLP-techniques for information extraction on informal text. To our knowledge there are no publicly available dataset containing genuine personal correspondence, so the Enron dataset and small dataset from Reddit has been used. Both consists of informally written text, however in correspondence in the Enron set is mostly business related and the Reddit posts are written for a public audience.

LIMIT To limit the scope of this project we cap the number of data-sources to a few common sources still including structured, semi-structured and unstructured data.

[3]https://spacy.io/
[4]https://www.nltk.org/
[5]https://scikit-learn.org/stable/
[6]https://nlp.stanford.edu/software/openie.html
[7]https://gate.ac.uk/family/developer.html
[8]https://cloud.gate.ac.uk/
[9]https://gitlab.com/ottr/lutra/lutra/

When managing and extracting personal information, the aspect of data protection becomes crucial. We leave this topic outside the scope of this project and inspired by Abiteboul et al., we propose that data aggregation, processing, knowledge extraction and storing is handled by the user's private server.

4.2 Ontology

The definition of a *Personal Knowledge Graph* PKG and its intended use as outlined by Balog and Kenter [2] is used as a starting point for defining the domain of the ontology. The requirements were further specified by listing questions we would want our DPA to be able to answer, if it had access to our own *Personal Knowledge Graph*. The questions are formulated with practical use in mind, without considering the technical aspect of extracting this knowledge and populating the graph.

4.2.1 Competency Questions

Questions formulated to capture the intended use of the ontology is called competency questions [54]. The scope, domain and requirements of our ontology is determined by these competence questions. It should be possible to use automated reasoning over the graph to answer these type of questions.

The competency questions are categorized into three categories: selection, binary and counting queries. Selection queries are questions that can be answered by returning a subset of values or entities from the graph. 'What is the number to my plumber?' is a selection query answered by returning *phone-number* of any company marked as *plumber* in the PKG, alternatively the *phone-number* to an entity named or labeled 'my plumber' in the PKG. A binary question will answer true or false based on a returned set. 'Do I know a plumber?' would return false if the query after *plumber* would return an empty set. Counting queries counts the number of entities returned by a selection query. The counting query 'How many artists do I like?' is answered by counting the set returned by the selection query 'What artist do I like?'

Using the examples given by Balog and Kenter[2] as a starting point we developed a set of competency questions envisioning an ideal PKG, and what practical use it might have. The list below shows a selection of the questions used to build our ontology.

Selection

- What is my mother's address?

- Give me direction to my mother from here?
- Where can I get a new drive belt for my scooter?
- Where did I buy my blender?
- What is Lina's phone number?
- Are there any interesting concerts next week?
- What is the number to my plumber?
- Where did we stay in London last time?
- What is the name of the hotel we stayed in last time?
- Give me the image of the cat my mother sent me.
- What did my sister think of the draft i sent her yesterday?
- Did mom answer?

Binary

- Do I know a plumber?
- Do I have any meetings today?

Counting

- How many people do I know?
- How many artists do I like?

Patterns

Competency questions define the scope of the knowledge represented in an ontology and is commonly used in ontology authoring techniques. Ren et al. describes a technique that revolves around using competency questions for both defining the scope and for validating the logic. The technique is intended for novice ontology authors.

The competency questions can be transformed down to a set of patterns representing all questions that are equal in structure. If we look at the noun phrase 'my mother' in the first question as a reference to a specific entity and 'address' a property, this question can be represented by the same template as 'What is Lina's phone number?', i.e. what is [instance] [property of instance]?. These patterns can be used to create SPARQL query templates and to define rule-based approaches for transforming natural language queries to SPARQL.

ID	Pattern	Example
1	What is [I1] [PE]?	What is [my mother's] [address]?
1	What is [I1] [ ] [PE]?	What is [my] [mother's] [address]?
2	What are the [I1] [PE] [I2]?	What are the [direction] [to] [my mom]?

4.2.2 Namespaces

Table 4.1 shows a list of namespace prefix bindings used throughout this thesis.

Prefix	URI
pkg:	http://pkg.org/pkg/0.01/ PKG Ontology
foaf:	http://xmlns.com/foaf/0.1/
schema:	https://schema.org/
relationship:	https://vocab.org/relationship/
personLink:	http://cedric.cnam.fr/isid/ontologies/PersonLink.owl#
rdf:	http://www.w3.org/1999/02/22-rdf-syntax-ns#
rdfs:	http://www.w3.org/2000/01/rdf-schema#
xsd:	http://www.w3.org/2001/XMLSchema#
pav:	http://purl.org/pav/

Table 4.1: Namespace prefix bindings

4.2.3 Included ontologies

FOAF: Friend Of a Friend

Friend Of A Friend [28] is a well known and widely used ontology developed for describing persons in the intent of facilitating a decentralized social network. This includes describing persons, organization, groups and objects, their properties and relationships. This ontology is created with the *Semantic Web* in mind and formalized using OWL.

RELATIONSHIP: A vocabulary for describing relationships between people

This ontology was created to define more fine-grained relationships then the generic *knows* relationship from FOAF. Every relationship that is defined in this vocabulary is a sub-property of the *knows* property defined in FOAF and having this property implies that the entity is a foaf:person.

PersonLink:The Person Link Ontology

The Person Link ontology by Herradi et al. [56] models rigorous and precise family relationships. It was created for a system helping people with memory loss and modelled to deal with incomplete and inconsistent data. It is included to provide a more expressive family model the other ontologies are able to provide.

Schema.org

Schema.org was created as an effort towards a shared vocabulary for structured data on the web. The vocabulary is maintained and updated as an open community process, backed by Google, Microsoft, Yahoo and Yandex. Schema.org is meant to provide a shared vocabulary for mark-up of web pages, making it possible for providers like Google, Microsoft and Yahoo to utilize published structured data directly. The ontology does not define logical rules beyond class structure, making it less expressive than the aforementioned ontologies.

Properties

Contact information In addition to the properties defined by the included classes we have included property values defined by the vCard standard. This is done to include more fine-grained contact information then specified by FOAF and schema.org and to facilitate inclusion and extraction of contact information. The vCard properties are defined as equivalent or sub property where possible. The mappings between properties defined in [57] is used, however the vCard ontology referenced in this web page is not.

4.2.4 The Personal Knowledge Graph ontology

The FOAF ontology was used as the starting point for building PKG ontology, it includes defined properties and relationships for entity types essential in an PKG. The RELATIONSHIP ontology is a natural extension of FOAF, including a rich vocabulary for expressing relationships between people, but the family relationships defined in this ontology is far less expressive then the PersonLink ontology. The PersonLink ontology is included by defining the equivalence mapping `foaf:Person` ≡ `PersonLink:Person`. It introduces two sub-classes defining gender in addition to a large set of family relationships with corresponding logic necessary for inference and verifying the family relationships. All relationships defined in RELATIONSHIP vocabulary are included as sub-property of the `foaf:knows` relationship. The family relationships defined by PersonLink is a sub-property of `pkg:relatedTo` which in turn is a sub-property of `foaf:knows`.

The FOAF ontology already define equivalence mappings from FOAF to schema.org; Person, Image (`schema:ImageObject`) and Document (`schema:CreativeWork`). The PKG ontology uses these classes in addition to a number of other from schema.org to describe entity types like EmailMessage, Product and Place.

Classes & Properties

The competency questions reveal the need for few additional entity types not defined by classes in the included ontologies. We add an entity type to express opinion expressed by anyone on anything in the graph. We include a class to describe directions, i.e. how to get from A to B. We include a class to describe visits, this class is based on THYMEFLOWs solution to describe the same data type.

Class - Visited Past tense visit defined as 'To go to see or spend time at (a place) with a certain intent.' [58]. In the *Personal Knowledge Graph* the entity type is used to describe the user's historical location data. The class references the location (schema:Place), start and end time.

Type	Name	Description
Class	pkg:Visited	$\texttt{pkg:Visited} \sqsubseteq \texttt{schema:Thing}$
Object property	pkg:placeVisited	$\exists\, \texttt{pkg:placeVisited schema:Thing} \sqsubseteq \texttt{pkg:Visited}$
		$\top \sqsubseteq \forall\, \texttt{pkg:placeVisited schema:Place}$
	pkg:visitor	$\exists\, \texttt{pkg:visitor schema:thing} \sqsubseteq \texttt{pkg:Visited}$
		$\top \sqsubseteq \forall\, \texttt{pkg:visitor foaf:Person}$
Data property	pkg:fromDateTime	$\exists\, \texttt{fromDateTime rdfs:Literal} \sqsubseteq \texttt{pkg:visited}$
		$\top \sqsubseteq \forall\, \texttt{pkg:fromDateTime xsd:dateTime}$
	pkg:toDateTime	$\exists\, \texttt{toDateTime rdfs:Literal} \sqsubseteq \texttt{pkg:Visited}$
		$\top \sqsubseteq \forall\, \texttt{pkg:toDateTime xsd:dateTime}$

Class - Direction are introduced as a subclass of `schema:howTo`. It allows the user to store information about how to get from point A to point b using external resources along with location data stored in the graph. The descriptions can be written step by step instructions or a link to a map depicting the instructions obtained from service providers like Mapquest, Google Direction, Bing Maps or OsmAnd.

Type	Name	Description
Class	pkg:Direction	pkg:Direction $\sqsubseteq$ schema:HowTo
		HowTo $\sqsubseteq$ CreativeWork
Object property	pkg:directionFrom	$\exists$ pkg:directionFrom schema:Thing $\sqsubseteq$ schema:Direction
		$\top \sqsubseteq \forall$ directionFrom Agent
	pkg:directionTo	$\exists$ pkg:directionTo schema:Thing $\sqsubseteq$ pkg:Direction
		$\top \sqsubseteq \forall$ directionTo Agent
	pkg:provider	$\exists$ pkg:provider schema:Thing $\sqsubseteq$ schema:CreativeWork
Data property	schema:url	$\exists$ schema:url DatatypeURL $\sqsubseteq$ schema:Thing
	schema:text	$\exists$ schema:text rdfs:Literal $\sqsubseteq$ schema:CreativeWork

Only properties and relationships used in our experimentation is showed in the table. The schema:HowTo allows detailed step by step descriptions and detailed metadata. The *schema:text* property is used to store the description and the *schema:url* is the link to the resource depicting the directions.

Class - Opinion are by definition 'a view or judgement formed about something, not necessarily based on fact or knowledge' [59]. The class is included to allow the user to stored expressed opinions by any person about any entity in the graph. This can be opinions expressed in conversation or written text about a topic. The class does emphasize a key distention between a PKG and a general KG, a PKG is a resource modelling the world from the user's perspective, not intended for sharing and knowledge within it represent the truth in the user's eyes only.

Type	Name	Description
Class	pkg:Opinion	pkg:Opinion $\sqsubseteq$ schema:Thing
Object property	pkg:hasOpinion	$\exists$ pkg:hasOpinion schema:Thing $\sqsubseteq$ foaf:Person
		$\top \sqsubseteq \forall$ pkg:hasOpinion pkg:Opinion
	pkg:about	$\exists$ pkg:about schema:Thing $\sqsubseteq$ pkg:Opinion
		$\top \sqsubseteq \forall$ pkg:about schema:Thing
	pkg:statedIn	$\exists$ pkg:statedIn schema:Thing $\sqsubseteq$ pkg:Opinion
		$\top \sqsubseteq \forall$ pkg:statedIn schema:CreativeWork
Data property	pkg:opinionPolarity	$\exists$ pkg:opinionPolarity rdfs:Literal $\sqsubseteq$ pkg:Opinion
	pkg:description	$\exists$ pkg:description rdfs:Literal $\sqsubseteq$ pkg:Opinion

Opinion polarity denotes if the opinion is positive, negative or neutral. This can be a the result of sentiment analysis or extracted from direct statements.

4.2.5 Provenance

While knowledge-bases generally consist of information that is relatively static, a personal knowledge-base will need to accommodate information that is transient in nature. Relationships, opinions and hobbies may change infrequently, but information regarding meetings, trips and dinners will be useful only in a short period.

Extracting information from conversational text has the potential to introduce contradictory knowledge into the graph. Unreliable, inconsistent and noisy data might be included in the graph if the accuracy of the IE methods are low.

These factors make it important to annotate the information with information about the source, temporal information and estimated confidence. To annotate the triples we use the light-weight provenance ontology PAV [60] and RDF-Star.

4.3 Building the Knowledge graph

In this section we will describe how we use existing methods and tools to model a user's contacts, calendar-entries, emails and other resources into one coherent graph structure. We focus on the methods used to extract knowledge from the private dataset, created from one users resources. Entities are given a identifier unique for the PKG where an appropriate global URI does not exist. The owner of the graph is identified as `pkg:User`.

4.3.1 Structured Sources

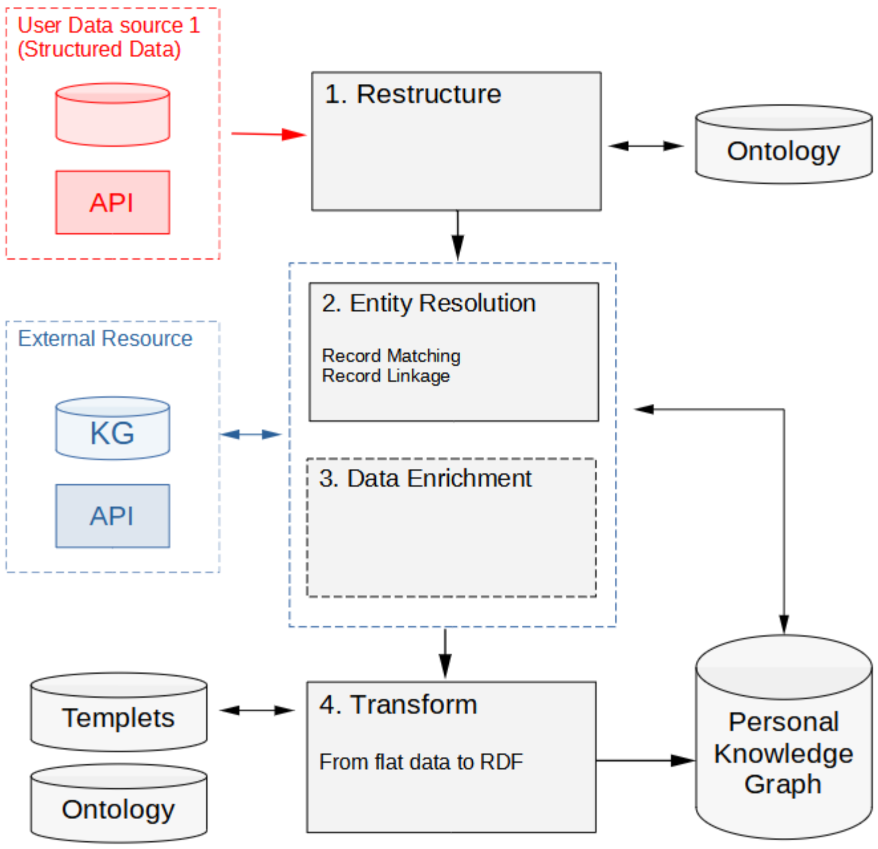


Figure 4.2: A simplified overview of the pipeline for knowledge-base enrichment from structured data sources. In this case the relationships between the user and the extracted entities are predefined

Short explanation of the pipeline in fig 4.4

- **1. Restructure:** Model the structured data from an external source to fit the ontology used in the PKG. Source and domain specific. The mapping is tailored to the provider's proprietary structure. The task of learning this mapping is not a part of this pipeline. (Ontology Mapping)
- **2. Entity Resolution:** The task of identifying entities cross multiple sources.
- **3. Data Enrichment:** Using an external source to add valuable information to the data. The external resource can be used to obtain global identifiers and URIs representing the entity on the web. e.g.
- **4. Transform:** Create triples from the extracted entities (the relationship between the user and the entities are predefined.) The step also ads metadata to the triples.

4.3.2 Contacts

Contact information is structured using established standards, it does not require much processing to enrich the graph with this data. However, when multiple sources is used, the challenge of entity resolution will need to be addressed.

The most common file format standard used to share contact information is vCard [RFC6350]. The format is recognized by most providers that handle contact information. There is an open standard client/server protocol called CardDAV [RFC6352], used to share and access contact information. This protocol is commonly used, however it is not uniformly supported by all providers, making it necessary to tailor synchronization using providers proprietary APIs.

In this project contact information is treated as static data stored in vCard [RFC6350] format. We do not investigate the technical challenges that may occur synchronizing contact information from multiple sources.

The Contact data result in two types of entities: foaf:Person and foaf:Agent.

1. Restructure

Mapping contact information extracted from vCard documents to the *Personal Knowledge Graph* is straightforward when the naming convention used for the properties in vCard is directly related to properties defined in the PKG ontology. A simple naive function based on [] was created to extract all common properties defined in the business cards.

2. Entity resolution

Contact information contain several properties that can be considered unique identifiers of person and Organization entities, like email address and phone number. These properties are therefore used to query the PKG before presiding to record linkage techniques for matching entities. To perform record linkage we used the Python Record Linkage Toolkit [61].

3.

Not relevant for contact information.

4. Transform

Example 4.3.1.

```
pkg:7f04a442-c920-46cd-afcf-9fdeb237327a
        vcard:adr            "Sandnesveien 911 SANDNES,  4312 NO" ;
        vcard:email          "melvin@norge.no" ;
        vcard:family-name    "Normann" ;
        vcard:fn             "Melvin Normann" ;
        vcard:given-name     "Melvin" ;
        vcard:nickname       "melmel" ;
        vcard:tel            "9112233221" .
```

Example 4.3.1 shows the RDF triples generated from a vCard serialized to turtle format. The entity URI is a randomly generated unique identifier used only in the PKG. The vCard in this example was created for the purpose of demonstration. The entity type is not directly stated, the class can be inferred from the properties. Example 4.3.2 shows metadata added to one of the triples from example 4.3.1 using RDF-STAR. The same method and basic information is added to all triples inserted in to the graph.

Example 4.3.2.

```
pkg:7f04a442-c920-46cd-afcf-9fdeb237327a vcard:nickname "melmel" .

<< pkg:7f04a442-c920-46cd-afcf-9fdeb237327a vcard:nickname "melmel" >>
        pav:createdBy    pkg:User ;
        pav:createdWith  "vCard import" ;
        pav:importedOn   "2021-05-10T16:03:34.816800+02:00"^^xsd:dateTime .
```

4.3.3 Location

Location data is included in the graph to provide valuable information about locations and their corresponding entities of interest to the user. This will allow the user to ask questions like "What is the phone-number to the hairdresser i visited last month?" and "How do I get to my next meeting?". People allowing location services on their phone will have access to their location history and rich information about the places they have visited, however this location data is contained within the providers ecosystem and only accessible through proprietary APIs for external applications.

The location data included is a data-dump from my own Google account. The extracted data is temporal information, location coordinates and an id of the location defined by google. A service providers out of the box solution was selected to simplify the task of identifying visits in the location data, excluding data points related to the travel from a to b. Organizations are identified based on proximity to the registered location, this means that the entities introduced to the graph via location data might be irrelevant to the user.

The location data result in 3 types of entities: schema:Place, pkg:Visited and schema:LocalBusiness. The entity type schema:LocalBusiness, a subclass of schema:Place and schema:Organization was used on Places identified as a business by map services.

1. Restructure

The data structures retrieved in the location data are proprietary. We use Google Location as an example and tailor the mapping from our source to the PKG ontology. Latitude and longitude are converted to decimal degrees and timestamps converted to DateTime. The recorded visit (pkg:Visited) is saved as a tabOttr file, converted using *Lutra* and inserted into the PKG.

2. Entity resolution

The Places and the organizations identified in the Location data needs to be resolved against the PKG. Because of geolocation data and unique identifiers, this can be done with high confidence.

3. Data Enrichment

We have experimented with retrieving additional information about visited location using both open source APIs from openstreetmap.org and Googles Places API.

Using the same service provider for data enrichment as the source of the location data makes extracting additional information about a place trivial, the proprietary identifier is used. Googles Places API uses its own vocabulary to describe the returned instances, the collected properties must therefore be mapped to the corresponding properties defined in the PKG.

As an alternative we query the overpass-api from openstreetmap.org using the transformed longitude and latitude values to obtain entities within a given radius. This results in a set of entities that will need to be filtered by type or by the identified Name provided in the location data. The API returns information about location tags(entities) using an ontology called OSMonto, this means this will also need to be mapped to the corresponding properties defined in the PKG.

Both services allow us to enrich our graph with information like opening hours, type of business, phone number and email, but the open source OpenStreetMap is dependent on community effort for expanding and maintaining their database. The data obtained from open sources like OpenStreetMap might have less coverage and a larger risk of including outdated information than popular solutions from large service providers.

4. Transform

Example 4.3.3 shows the OTTR template in stOTTR format used to expand instances of schema:Place from location data stored as a tabular tabOttr [52] file. Example 4.3.4 shows the resulting triples generated by Lutra.

Example 4.3.3.

```
pkg:Geo[ottr:IRI ?geo, xsd:string ?lat  , xsd:string ?lon] :: {

    ottr:Triple(?geo,rdf:type,schema:GeoCoordinates),
    ottr:Triple(?geo,schema:latitude, ?lat ),
    ottr:Triple(?geo,schema:longitude, ?lon)

}.

pkg:Place[ottr:IRI ?place, xsd:string ?id_, xsd:string ?lat,
                            ... xsd:string ?lon, xsd:string ?name] :: {
```

```
        ottr:Triple(?place,rdf:type,schema:Place),
        ottr:Triple(?place,schema:name, ?name),
        ottr:Triple(?place,schema:identifier, ?id_),
        ottr:Triple(?place   ,schema:geo , _:blank),
        pkg:Geo(_:blank,?lat,?lon)
    }.
```

Example 4.3.4.

```
pkg:a3b7b2bc-ecc0-4409-8246-2584738b8641
        a                  schema:Place ;
        schema:geo         [ a                 schema:GeoCoordinates ;
                             schema:latitude   "58.9374504" ;
                             schema:longitude  "5.6952882"
                           ] ;
        schema:identifier  "Google:ChIJ6xUQan41OkYR1h4p33wZB_g" ;
        schema:name        "Kitty Kiellands Hus" .
```

Additional information retrieved from Googles Places API are inserted in to the PKG using googles propitiatory identifier to match records and a simple function to restructure the data to triples.

4.3.4 Streaming services - music

Information from streaming services is included to the PKG to enrich the graph with information about the user's preferences. This will allow the user to use information about entities of interest normally contained within a provider's ecosystem. This along with location data and a service like *songkick* allows the user to ask questions like 'Are there any interesting concerts next week?'.

Spotify API is used to obtain name, artist and album recently listened to.

1. Restructure

The data structures retrieved from streaming services are proprietary. We use Spotify as an example and tailor the mapping from our source to the external resource used for Music. Enabling extraction form other streaming services would require additional mappings.

2. Entity resolution

Artist and album entities in the graph have multiple unique identifies, ranging from global ids like ISRC, ids in external knowledge graphs to the proprietary id from the streaming service. Artist entities not yet found in the PKG, will need to be matched against records in the external resource. Detailed information from the streaming service allows us to match records with high confidence. Artist name, Album and song is used if the ISRC is not found in the external resource or obtainable from the streaming-service.

MusicBrainz contain the URI identifier for artist from a few popular streaming services like Spotify and Apple Music, however not all artists have these.

3. Data Enrichment

MusicBrainz is used as a external resource to complement the graph with global identifiers and links to relevant resources for the artiest of interests.

4. Transform

Example 4.3.5.

```
<https://musicbrainz.org/artist/61ed9c9c-79eb-4e8f-8015-bd599ac0ab49>
        a                  schema:MusicGroup ;
        schema:genre        "alternative metal" , "gothic metal" , "metal"  ;
        schema:identifier  "000000010660513X" ;
        schema:name        "Katatonia" ;
        schema:sameAs
            <https://www.wikidata.org/wiki/Q515273> ,
            <https://www.songkick.com/artists/58766>,
            <https://www.allmusic.com/artist/mn0000855252>
```

Table 4.3.5 shows an shows the RDF triples generated for an artist entity serialized to turtle format. The property value schema:genre and schema:name is from the streaming service, the ID, schema:identifier and the urls are from musicbrainz. The URIs shown under schema:sameAs is selection from a total of 22 in this specific case.

4.3.5 Unstructured and semi-structured Sources

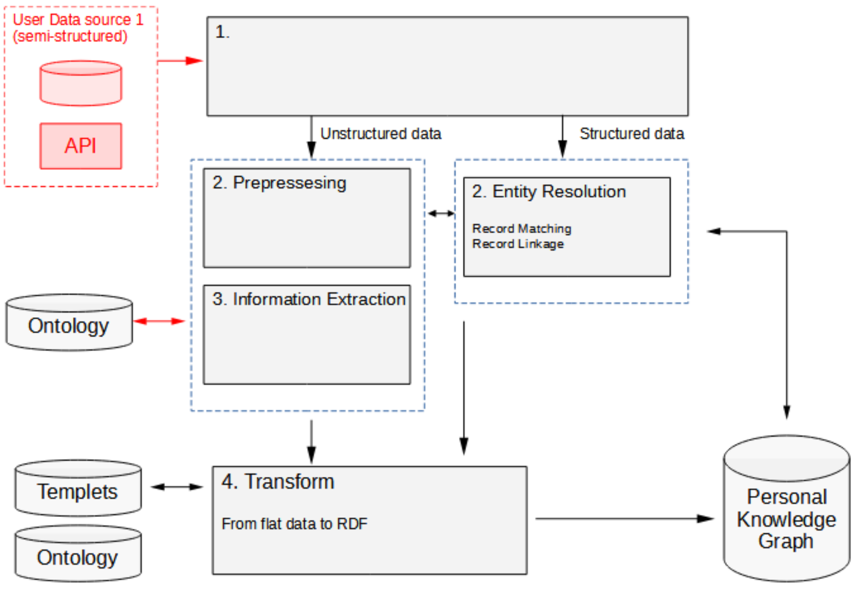


Figure 4.3: A simplified overview of the pipeline for knowledge-base enrichment from semi-unstructured data sources

4.3.6 Calendars

The calendar data imported uses the iCalendar [RFC5545] specification.

The Calendar data result in 2 types of entities: schema.org:Event and schema.org:Schedule (Calendar entry with recurring rule). The implementation is minimal, including only basic information about meetings. IE is not performed on calendar entries.

Example 4.3.6.

```
pkg:cc1dd9da-6426-450f-a093-c4171dc3fde0
        a                    schema:Event ;
        schema:about         "This  is about something something" ;
        schema:attendee      "mailto:demo@gmailmail.com" ;
        schema:attendee      pkg:User ;
        schema:doorTime      "2021-05-24T06:00:00+00:00"^^xsd:dateTime ;
        schema:endDate       "2021-05-24T07:00:00+00:00"^^xsd:dateTime ;
        schema:identifier    "6puyiio5681@google.com" ;
        schema:name          "This is a meeting" ;
        schema:organizer     "mailto:demo@gmailmail.com" .
```

Example 4.3.6 shows the RDF triples generated from a calendar entry serialized to turtle format. The entity URI is a randomly generated unique identifier used only in the PKG.

All details of this meeting have been changed. The organizer of this meeting did not exist in the PKG, and therefore represented by the email.

4.3.7 Email

Personal emails contain a wealth of information about its user. In this project we aim to extract as much relevant personal information from emails as possible, utilizing the standardized structure and using NLP techniques to identify entities and the relationships between them.

Both the dataset used for **Framework & knowledge structure** and the dataset created for **Information Extraction** contain email data, however the pipeline as described is only used on the private dataset. This is done because we want to evaluate how our approach benefits from a graph containing entities relevant to the owner of the email .

We use SMTP to obtain emails from two separate email providers from the same user. We use Python *emaplib* for receiving emails and Python *email* package to decode them.

Emails are mapped to schema.org: EmailMessage and any attachment (schema.org: messageAttachment) to schema.org: CreativeWork. IE on the text may result in instances from any class or relationship defined in the PKG.

Headers

We consider only headers defined by the standard, not including proprietary X-headers. The fields processed and included in the graph are:To, From, Subject, Message-ID and Date.

Message-ID: is combined with the base url of the graph and used as the URI for the EmailMessage entity.

Date: string is converted to *Date Time* and used as schema:dateSent on the EmailMessage entity.

Email address are considered identifiers for entities in the PKG. Generic and role-based Email addresses from organizations should therefore represent the Agent/organization entity not the person using it. If the email address does not exist in the PKG, the address is further processed to try to disambiguate it against the PKG.

$$\text{From: } Name \ < username@domain.com >$$

1. Name field , 2. split username 3. domain 4.

From The resolved entities are linked to the EmailMessage by schema:sender in the graph.

To: The string is split into a list and each entry processed individually. The resolved entities are linked to the EmailMessage by schema:recipient in the graph.

Subject The subject field may contain references to other entities in the graph such as persons, meetings locations. The string is therefore cleaned before using a rule-based entity recognition pipeline from Spacy 4.3.8, where names and nicknames of entities in the PKG are represented as patterns.

Content

The content of emails can be split into multiple parts, where each section is described by its own header. We use the defined content type to distinguish between three types of data 'text/plain', 'text/HTML' and attachments. Each of these data types is processed differently.

Attachments are downloaded and saved with a reference to the original email, but we do not perform any IE on the content of the document, nor do we aim to find other properties like author. The documents are represented as entities of type schema:CreativeWork.

Body text/HTML In addition to being used to make changes to emails visually, HTML can contain structured information. This is commonly used on automatically generated emails like order confirmations and reservations. Text 'text/HTML' content is therefore run threw an extractor (python extruct[10]) to identify any *microdata*, *JSON-LD* and *Microformat*. If the section contains structured data of this type, the extracted information is saved, if not, the HTML is cleaned and processed in the same way as plain text sections.

Body text/plain In addition to being more informal in nature, the text itself has a common structure normally not found in the corpuses used to train publicly available pretrained NLP-models. These models are designed for clean text, not to manage signature blocs and greetings, resulting in the need to separate and classify sections of text. Separating these sections reduce errors in *sentence segmentation*, a small but essential part of the NLP pipeline. The common structures used in signature blocs allows for contact information to be extracted with simple rule-based approaches. Preforming

[10]https://pypi.org/project/extruct/

Information Extraction on correspondence require knowledge about the author and the intended reader to correctly 'disambiguate' personal pronouns found in the text. For this reason, forwarded emails and reply chains sections needs to be separated and processed differently.

Email bodies commonly consist of a greeting/salutation, the text and a signature.

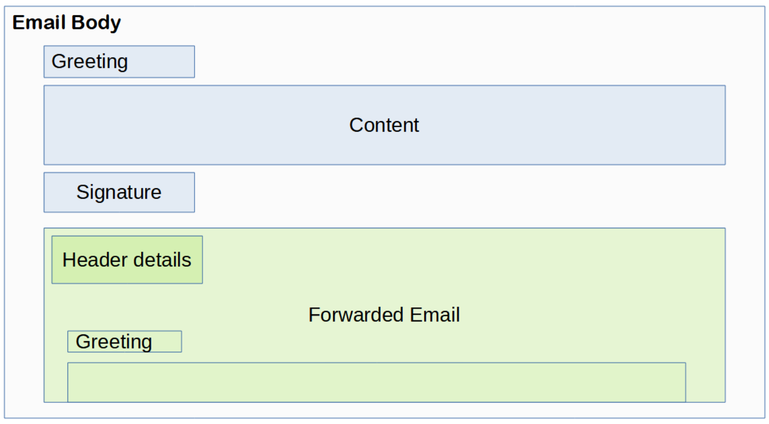


Figure 4.4: text/plain

- **Cleaning** Removing HTML tags using Beautiful Soup [62]
- **Splitting sections**: Conversation threads are identified and separated using id references and the 'In-Reply-To' field in the header, but separating and processing forwarded emails from a third party is a lot harder. There is no standard depicting forwarded email and how to include header information from the original email. For this reason, we split and separate forwarded content from the incoming email, but do not try to reconstruct and extract knowledge contained within the original email.
- **Greeting/salutation**: Rule-based approach to identify/confirm name/nickname of the receiver. A naive approach using Spacy Matcher.
- **Content**: Information Extraction using a Spacy pipeline described in 4.3.8
- **Signature**: Signatures can contain useful contact information and the commonly used structure of these signature blocks makes it possible to identify and extract this information using rule-based approaches or ML models. We use the Python package Talon [63] for the Enron email dataset. This is not a viable option for our private dataset that are written in Norwegian. We therefore use a simple naive rule-based approach where we try to identify signatures containing contact information. We simply calculate the fraction of named entities like phone numbers, addresses, emails in a text bloc. The entities are identified using patterns (entity_ruler) and the text blocs are identified by white space separation.

4.3.8 NLP pipeline

The emails in the personal dataset is predominantly in Norwegian, we therefore use a pretrained Norwegian pipeline created for spaCy [64].

$$Tok2vec \rightarrow Morphologizer \rightarrow Parser \rightarrow NER \rightarrow Lemmatizer \rightarrow Entity \quad Ruler$$

The Entity Ruler contains patterns of the names for all named entities in the PKG, along with patterns to identify contact information like phone numbers and emails. The entity ruler overrides the NER.

The method is limited with no co-reference resolution, and reliant on hand built patterns using a dependency matcher for identifying relationships.

English pipelines

The language limitation and the small size of this private dataset makes it unfit for evaluating NLP pipelines for the purpose of IE on informal text. We therefor experiment with publicly available dataset for IE on informal text in Chapter 5

4.4 Using the Personal Knowledge Graph

Based on the patterns defined by the competency questions, we built a collection of SPARQL-templates. The main objective is to determine if the graph structure allows for the questions to be answered correctly using SPARQL, not focusing on how these natural language questions can be interpreted and applied to the appropriate SPARQL-query. We also aim to make the SPARQL-queries as uncomplicated as possible.

Example 4.4.1 shows a SPARQL example to answer the basic question 'What is my mother's address?'. The `pav:importedOn` relationship is used to get the most recently asserted address. The property `personLink:1.3.1` is the identity of the mother of relationship in the PersonLink ontology. The ontology is multilingual and uses labels to provide semantically meaningful names for object properties. Example 4.4.2 shows an example of using the labels to obtain the relationship, however using this methods requires RDFS labels to be unique.

Example 4.4.1.

```
SELECT      ?subject ?address ?imported
WHERE
    {
      ?subject  personLink:1.3.1    pkg:User .
      ?subject  vcard:adr           ?address .
      << ?subject  vcard:adr  ?address >> pav:importedOn ?imported
       }

ORDER BY DESC(?imported)
LIMIT 1
```

Example 4.4.2.

```
SELECT      ?subject ?address

WHERE
    {
      ?rel rdfs:label  'MotherOf'@en .
      ?subject ?rel pkg:User .
      ?subject vcard:adr  ?address.
    }
```

Chapter 5

Experiments and Results

In this chapter we evaluate our proposed knowledge structure, our approach towards populating the graph and its practical use. A PKG is the result of accumulating and extracting data from multiple disparate data sources obtained from one user. To give a quantifiable evaluation of our approach end-to-end, we would need personal data from multiple users or a group of willing participants and a large-scale user evaluation is beyond the scope of this project. Because of this and the fact that the objectives reach over multiple tasks within different research areas, each part of our approach is evaluated separately.

In Section 5.1 we describe findings from our preliminary research into datasets containing informally written text suitable for IE given the domain defined by the PKG. In Section 5.2 we present our finding on IE on informal conversational text using a public dataset.

5.3 presents our result from building our own PKG using our own data and the methods described in Chapter 4.3, what we were and what we were not able to extract and include into our PKG. Lastly, we evaluate the practical use of the PKG, both with a fully populated graph where all knowledge necessary answer the queries are present and an end-to-end evaluation, where only entities extracted from our data is included in the graph.

5.1 Personal information in informal text

In this section we will look at available datasets containing informal text and aim to find out how much knowledge relevant in the context of a PKG is contained within these datasets. We do this by annotating a random selection of documents with a class based on relevance. We first look at a subset of the Enron dataset [65], secondly we look at a selection of public blog posts and accompanied comments from Reddit.

Irrelevant documents are defined as irrelevant where nether the subject nor body of the text contain information, named entity mentions or relationship useful for the PKG. IE if there is nothing to gain by preforming IE on the text.

Relevant - Suitable for NER documents are defined as relevant and suitable for NER where the subject or body of the text contain entity mentions of entity types defined in the PKG.

Relevant - Suitable for RE & NER documents are defined as relevant for RE where the subject or body of the document contain knowledge that might be useful for the PKG. If there is enough information either inferred or directly stated to extract a triplet conferring with the vocabulary defined by the PKG the document is in this class.

Type	Source	Standard	Instances
email	Enron Dataset [65]	RFC822	385
Blog-posts	Reddit		100

Table 5.1: Instances used to test

Enron Dataset

The Enron Email dataset does consist of a large collection of informally written text suitable for evaluating NER, co-reference resolution and entity linking on informal text, however these are for the most part business related correspondence. The Enron dataset has been used extensively in research, from different classification task, NER [66], EL [67] and Knowledge Extraction for social graphs. Multiple techniques and models to leverage the structured nature of emails has proven to identify and extract knowledge from email signatures [63] and header with good accuracy. We focus on the conversation itself, to see if we can extract personal knowledge beyond the contact information found in the signature and the data in the header. For this reason we assess the content of the text only for the purpose of evaluating IE models in Section 5.2.

	Description	nr
Irrelevant	Emails with no information relevant for the PKG.	228
Relevant NER	Emails with entity mentions	115
Relevant RE (PKG)	Relevant emails with personal relationship between two entities	42

Observations The relevant emails found in this sample set contained mostly contact information and references to meetings. Two personal relationships were revealed, one 'relative' (inferred) and one 'Aunt' (noun-phrase in signature). One birthday was mentioned.

Data from Social platforms

With scarcity of data containing personal correspondence, the majority of NPL-research focused on informal text uses blog posts and public discussion sites as source [26, 32]. Popular services like Reddit contain large collections of discussions in a wide range of subjects in multiple languages, although predominantly in English. We wanted to know if this type of data could be close enough in nature to be used as proxies and if relevant personal information is offered up by its users and in what form.

We used the Reddit API to obtain 100 random post and their direct comments from random sub-threads. Not including comments on comments. The posts are classified in the same manner as the Enron emails. 15 of these post where empty. In total 363 comments.

	Description	nr
Irrelevant	No information relevant for the PKG.	40 (15)
Relevant NER	Posts with entity mentions	44
Relevant RE (PKG)	Relevant personal relationship between two entities	16

Observations The relevant information found among these comments and post were a good portion of opinion on named entities like products and companies. Mentions of hobbies, pay, occupation, family members and age where revealed. The threads provided context and most of the relevant information in these texts where directly stated.

5.2 Information extraction

In this section we will present our findings and observations from experimenting with publicly available NLP-pipelines on public datasets. The dataset used is a selection of emails from the Enron [65] dataset. Even though this dataset has a scarcity of relevant relationships within them as shown in Section 5.1, it is to our knowledge the only publicly available dataset containing personal correspondence resembling text sources relevant for a PKG.

5.2.1 Experimental Setup

We use the Enron emails classified as suitable for RE & NER to evaluate how publicly available NLP-models interpret informal text. This approach will not give a quantifiable evaluation of the models used, but by studying each occurrence individually we aim to find out where models trained on formal text fail and if there is anything that can be done to improve the results.

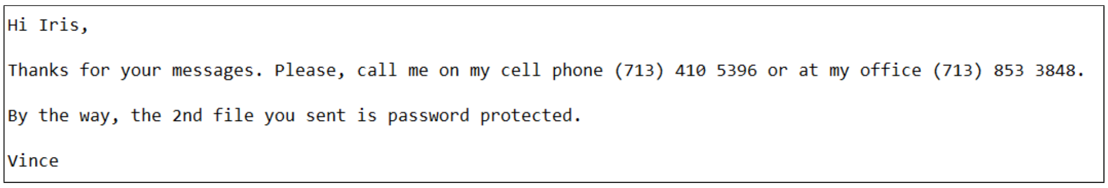
We use visualisation tools, *GATE* Annie and *Spacy's* displaCy when we evaluate each email. We focus on the text where relationships are mentioned or inferred, how these models preform on sentence-segmentation, POS-tagging, NER, dependency parsing, noun-chunking and co-reference resolution, i.e. how the models interpret the informal text and if it is possible from this to identify the knowledge within them. The evaluation of syntactic features is limited due to the authors sparse knowledge within linguistics.

The emails are cleaned versions where the signature and forwarded sections are ignored. The out of the box signature extraction tools used are not evaluated here.

5.2.2 Result

Example 1:

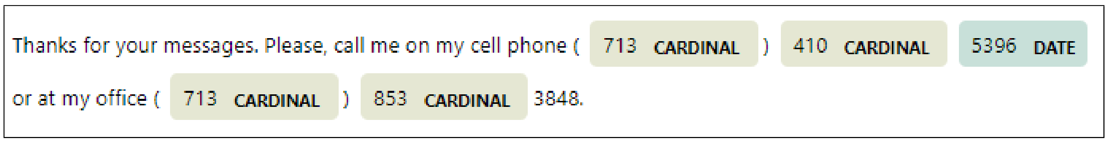
The first example shown in Figure 5.1 represent the less complex examples found among our set of emails containing relevant information. The sentences are short, concise and properly punctuated. The target information here is the phone numbers, typically found in the signature. Signature sections are not run through these NLP-models, but because the information occurs in the text and it is representative for the type of information that is easy to identify, we include it as an example.



```
Hi Iris,

Thanks for your messages. Please, call me on my cell phone (713) 410 5396 or at my office (713) 853 3848.

By the way, the 2nd file you sent is password protected.

Vince
```

Figure 5.1: The first example email.

spaCy Default Pipeline Figure 5.2 shows the NER result from a pretrained pipeline from spaCy, trained on a large web dataset consisting of blogs, news articles and comments. Figure 5.3 shows dependency parsing including noun chunking from the same model.



Thanks for your messages. Please, call me on my cell phone (713 CARDINAL) 410 CARDINAL 5396 DATE or at my office (713 CARDINAL) 853 CARDINAL 3848.

Figure 5.2: The figure shows the result from spaCy NER-model using a pretrained model with no modifications.

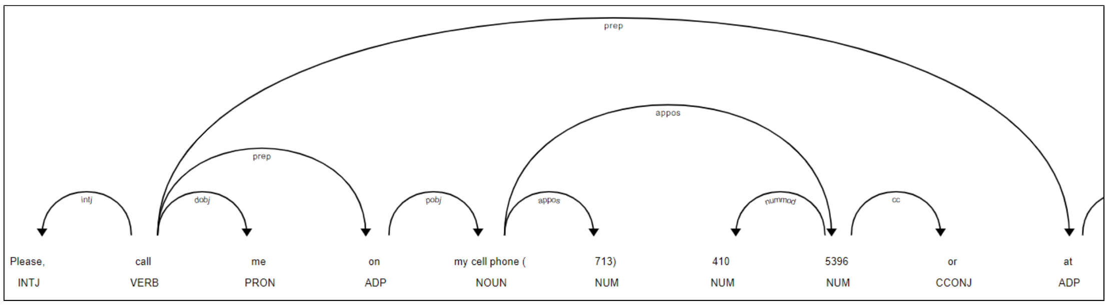
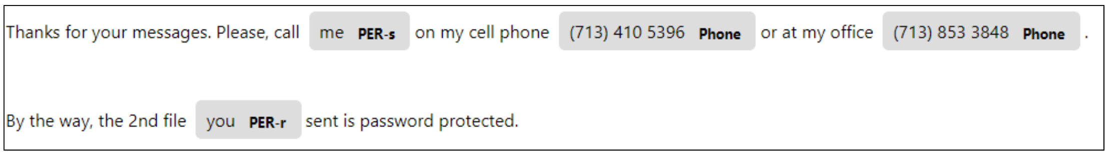


Figure 5.3: The figure shows the result from spaCy dependency parser using a pretrained model with no modifications.

spaCy Pipeline with entity ruler The output from 5.2 and 5.3 demonstrate that pretrained models do is not directly applicable beyond their intended use. However, it is also clear that the main issue here is that not all entity types relevant for the PKG is included in the NER-model used in the pipeline. Figure 5.4 and 5.5 shows the same sentence run thru the same base model with an added entity ruler. This simple modification lets us identify the sender and receiver mentioned in the text by personal pronouns, it also captures property values by recognizing patterns. By merging the recognised patterns and noun-phrases, the dependency parser now have a clearer result. Rule based relation extractor. The entity ruler is limited to a small set of manually created patterns.

$$
\begin{aligned}
&Tok2vec \rightarrow Tagger \rightarrow Parser \rightarrow NER \rightarrow Attribute\quad Ruler \\
&\qquad \rightarrow lemmatizer \rightarrow Entity\quad Ruler \qquad (5.1)
\end{aligned}
$$


Thanks for your messages. Please, call me PER-s on my cell phone (713) 410 5396 Phone or at my office (713) 853 3848 Phone .

By the way, the 2nd file you PER-r sent is password protected.

Figure 5.4: The figure shows the result from spaCy NER-model using a pretrained model with an added entity ruler

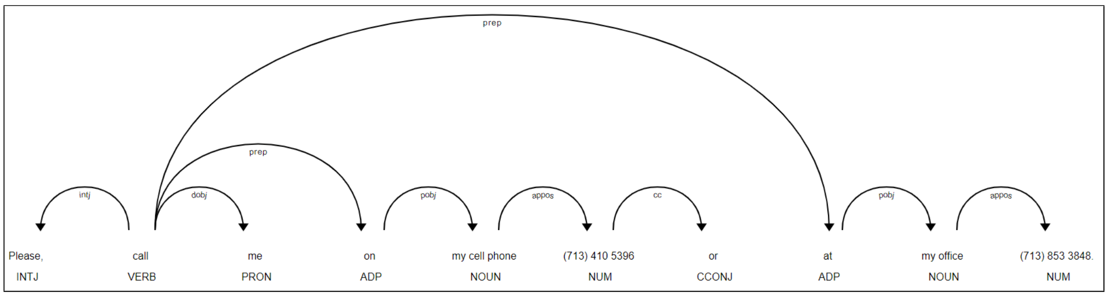


Figure 5.5: The figure shows the result from spaCy dependency-parser using a pretrained model with an added entity ruler

GATE Annie Figure 5.6 is from GATE Developer showing the result from a default Annie pipeline. The pipeline uses both a gazetteer and rules, and is therefore able to identify the numbers as phone-numbers. The gazetteer allows the model to identify common names, including identifying gender in addition to separating first and last names.

$$
\begin{aligned}
&Tokinizer \rightarrow Gazatteer \rightarrow SentenceSplitter \rightarrow POS\quad tagger \rightarrow \\
&\qquad NE\quad Transducer \rightarrow OrtoMatcher \rightarrow Coreferencer \qquad (5.2)
\end{aligned}
$$

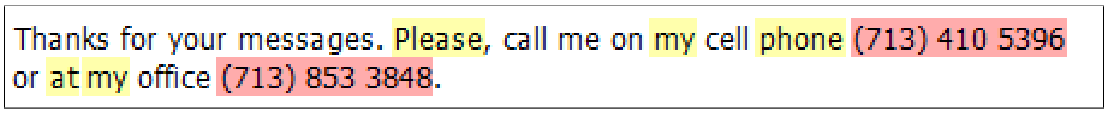

Thanks for your messages. Please, call me on my cell phone (713) 410 5396 or at my office (713) 853 3848.

Figure 5.6: The figure shows the result from Gate Annie pipeline

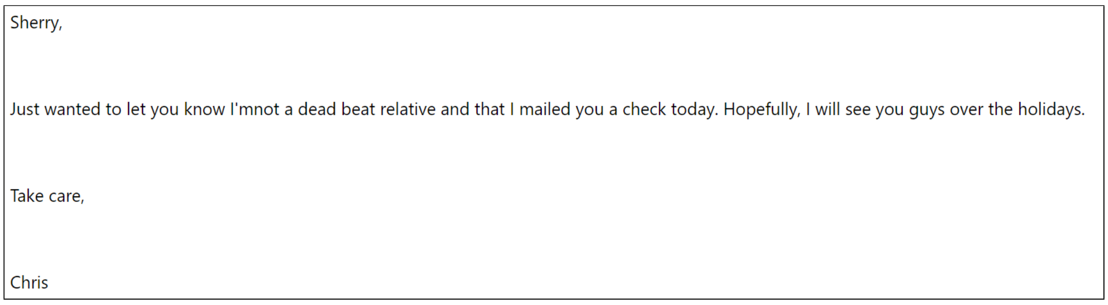
OpenIE The table below shows the result from Stanford openIE on example 1. openIE is an open relation extractor that relies on syntactic features to extract binary relationships in an open domain. The model is not able to detect any binary relationships in the sentence containing the phone-number, nether as is or in the case where the phone-numbers are chunked together to single tokens. The two triples it did detect do not contain any relevant information, nor did it interpret the sentence correctly. These types of errors are easy to filter out due to the lack of reference to any entities.

Subject	Predicate	Object
2nd file	is	By way protected
file	is	By way protected

Similar examples within the sample set of relevant emails shows that while the entity ruler are a reliable method for identifying common patterns like phone numbers, it was not possible to reliably identify the entity they belong to. Creating hand-built patterns for identifying these relationships using the syntactical features is feasible, but the number of pattern would be nearing one per mention in our small sample sett. The informal nature of these texts are a large contributor to this. Basing a model on the features identified by these pipelines is possible, however it would require a lot more data then we where able to annotate.

Example 2:

The second example shown in figure 5.7, is one of two emails where family relationships where disclosed among our sample set. The relationship is not directly stated and falls under the general related_to, from which all family relationships defined in our ontology are descendants. Figure 5.8 and 5.9 shows the results from the spaCy pipeline 5.1, NER and dependency parsing respectively. While inferring this relationship is not possible, the example shows the common problem of typos causing the POS tagger and dependency parser to incorrectly interpret the sentence.


Sherry,

Just wanted to let you know I'mnot a dead beat relative and that I mailed you a check today. Hopefully, I will see you guys over the holidays.

Take care,

Chris

Figure 5.7: The second example email.

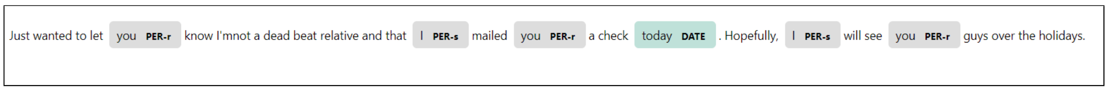

Just wanted to let you PER-r know I'mnot a dead beat relative and that I PER-s mailed you PER-r a check today DATE . Hopefully, I PER-s will see you PER-r guys over the holidays.

Figure 5.8: The second example NER annotation spaCy 5.1

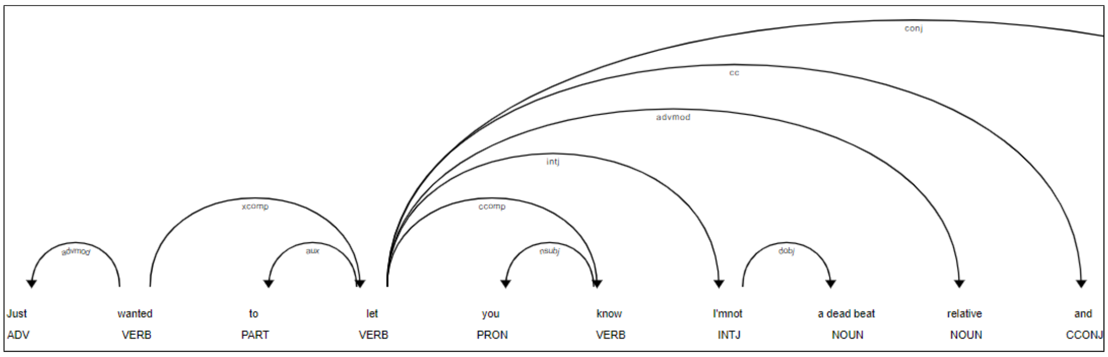


Figure 5.9: The second example DEP, POS spaCy 5.1

5.3 Building the knowledge graph

In this section we will show the results from building our PKG, what we were and what we were not able to extract and include into our PKG. Table 5.2 shows an overview of the records used to build our PKG. All the records are personal data belonging to one user, three calendar entries and four contact records where created for the purpose of demonstration.

Type	data	Standard		Instances
email	static		RFC5322	3127
Calendar	dynamic read only	iCalendar	RFC5545	5
Location Data	static			4 690
Contacts	static	vCard	RFC6350	17
Streaming services	dynamic read only			50

Table 5.2: Records used to build the PKG

5.3.1 Result

Our location data contained a total of 933 places from 4600 visits, where 407 of these where Local Businesses. Our contact data contained 17 persons where only 5 of them included more then a phone number and a name. The calendar consisted of only 5 entries where most of them where fabricated for the purpose of testing. The Streaming services in the final run contained the names of 50 artists.

Email

The mailbox used for this is from the same user as the rest of the data sources, to allow to use the PKG as a tool to identify relevant mention in the email. This is a small set of emails predominantly in Norwegian, this meant that we where limiting us to spaCy for pretrained models and that we had to add patterns to our Entity Ruler. It was not possible to create a unbiased test set from this private data.

The email set contained 3127 emails from 521 unique email addresses within 308 domains. For 342 of these email addresses it was possible to obtain name of the sender, person or organization. The majority of emails in this set is automated responses and newsletters.

$$Filter \rightarrow Clean \rightarrow Sectioning \rightarrow Classify \quad Signature/Text \tag{5.3}$$

Information Extraction IE on our private emails resulted in identifying some entity mentions and some contact information, however the experimental setup and the amount of noise this process generated led us to not include it into our PKG.

- **Filtering:** To limit the amount potential noise extracted by the IE, we filter out emails that are thought to be automated emails like newsletters. Our naive approach did not remove all automated responses, but newsletters where, and no relevant correspondence was ignored. The filter just decides what to process for IE, the emails are still preserved in the PKG. 2040 emails where filtered out, this is 65% of the set.
- **Cleaning:**
- **Signatures:** The method we used did not perform adequate,identifying only tiny fraction of signatures containing contact information.

- **spaCy IE-pipeline:** We were not able to extract any relationships directly from the text, but we were able to identify mentions of entities from our PKG in the emails and detect simple patterns like phone numbers and dates.
- **Attachments:** 239 documents and images, from 60 named entities.

Embedded metadata in HTML email content

The Python package *extruct*, where used to identify and extract embedded metadata in HTML content from emails. The package is capable of detecting *Microdata*,*Jason-ld*, *Microdata*, *rdfa* and open-graph. We do not have statistical data on how common the use of structured data is in automated emails-response and Newsletters, we only have access to one email box with emails from a relatively small set of organizations. Out of 3127 incoming emails, structured data where detected in 476 of them. However, only 100 of them included relevant data using a known structure. These where reservations described using *Microdata* and the schema.org vocabulary and contact-information embedded using *Microformat* and h-card standard (embedded Vcard). This is information relevant for the *Personal Knowledge Graph*, easy to extract and include.

Dataformat	Vocabs	nr	Providers
Microdata	schema.org/TrainReservation/Person/Organization ++	33	2
JSON-LD	schema.org/EmailMessage/ViewAction/	2	
Microformat	h-card	65	
RDFa		394	

Data Enrichment

We used domain specific sources to obtain additional information about entities found in our structured data. MusicBrainz was used for entity-type Artist and for organizations (local businesses found in our location data) OSMonto and Googles Places.

The external sources enabled us to interlink artist we like with public knowledge graphs like dbpedia, obtain links to relevant web pages and get global identifier for the artist. All artist in our dataset where fond and linked correctly to MusicBrainz, very few of them where not a perfect match and they all had relevant resources in MusicBrainz.

Obtaining additional information about places visited using OSMonto and Googles Places, did allow us to obtain contact-information, type of business, opening-hours and links to relevant web-pages. The service used to obtain the location data already provides all this information in an easily accessible way, however extracting this information allowed us to use the information outside the ecosystem. From our testing OSMonto contained some information that was found to be outdated.

5.4 Using the knowledge graph

In this section we will evaluate the usability of our PKG both given the limitation of the data we were able to extract from our personal data sources and with a fully populated PKG using the competency questions defined in Section 4.2. This will also evaluate the ontology used relative to the use cases (queries) defined.

In Section 5.4.1 we show queries covered by the ontology, these are queries that is possible to answer using SPARQL given that the knowledge is present in the PKG. These are in term divided in to three categories, queries that are answerable, where the information is missing in the data sources and where we where the information is represented in the sources but where we are not able to extract this knowledge.

In Section 5.4.2 we show the queries that are not covered by the ontology. These questions are not possible to answer using the ontology as is.

5.4.1 Covered by the ontology

The table below shows some examples of the types of questions we were able to answer without adding additional knowledge to our PKG manually.

- **1. What is [my mothers] [address]?:** The first pattern is obtaining a property value of any named entity. The SPARQL query written to answer this is limited to knowledge within the PKG and exact matches on name/nickname of the entity. The noun phrase [my mother] did not occur as a name or a nick in our PKG, however the PKG due contain a large number of entities and properties that can be obtained using the graph.
- **2. Do i have any meetings today?**
- **3. How many artists do I like?**
- **4 Get me the document Anne sent me.** An example SPARQL query for retrieving documents by using entity name is shown in Example 5.4.1.

Example 5.4.1 shows the last example in the list. The query is very open not specifying type of message, nor when the document was sent. Using only a first name might lead to ambiguity if multiple users has the same name. The query simply returns the last documents sent by an entity with first name "Silje" sorted by date. In our testing the document names of message attachments are stored as literals `schema:messageAttachment`. schema:EmailMessage a subclass of schema:Message.

Example 5.4.1.

```
SELECT      ?id ?attachment ?sender ?sent
    WHERE
        {
            ?id  rdf:type schema:Message     .
            ?id  schema:recipient  pkg:User  .
            ?sender foaf:firstName|schema:name "Silje" .
            ?id schema:sender ?sender .
            ?id schema:messageAttachment ?attachment .
            ?id schema:dateSent ?sent
        }
ORDER  BY DESC(?sent)
LIMIT 3
```

The queries we were not able to answer, falls into two categories, firstly and by far the most common in our experiments is missing knowledge, secondly are queries we where not able to answer due to the limitation of the ontology itself. Missing knowledge is due to two reasons, missing information in the users sources and failure of extracting structured knowledge.

Missing data

Using the PKG without enriching the graph with key information limited the practical use of the PKG. This was largely due to the fact that a good portion of the questions we had defined are dependent on knowing the user's relationships to the persons found in the graph.

- **1. What is [my] [mothers] [address]?:** No personal relationships were mentioned in the personal data, and thus no relationships of this kind possible to extract. To answer this query we would need to insert the first triplet shown in Table 5.3 to the PKG. A query to answer this simple question is shown in Example 4.4.1.

- **2. Give me [direction] [to] [my] [mother] from here?** This query is dependent on knowing where the user is, who the mother is, her address and a service capable of providing directions. A simple workaround using external services and the addresses only is shown in Example 5.4.2. This example require the MotherOf relationship to be established.
- **3. What did my sister think of the draft i sent her yesterday?** This type of opinions did not occur in the dataset, nether would we be able to detect this type of information using the current IE pipeline. It would require sentiment analysis

Table 5.3 shows manually inserted triples to define family relationships.

Subject	Predicate	Object	Predicate label
pkg:9f46af3c-...	personLink:1.3.1	pkg:User	MotherOf
pkg:9f4df6dc-...	personLink:1.3.2	pkg:User	FatherOf
pkg:9f430586-...	personLink:5.1	pkg:User	SisterOf

Table 5.3: Manually inserted triples to define family relationships

Example 5.4.2.

```
SELECT  ?id ?name ?directionTo
    WHERE
        {
        ?subject   personLink:1.3.1 pkg:User .
        ?subject   schema:address ?address .

                BIND(URI(CONCAT(
                        'https://maps.google.com?saddr=Current+Location&daddr='
                        ,REPLACE(?address, "[ ]","+" ))) as ?directionTo)
           }
```

Unextractable data

There were a few types of information that were present in the data-sources, but not possible for us to extract.

- **1. Where did [I] [buy] [my blender]?** Information about where, what and for how much a large number of items where bought for were present in the emails, either directly listed in confirmation emails or more commonly in the form of PDF attachments. While we did not have any blenders among the items, the query is representative.

- **2. Give me the image of the cat my mother sent me.** This query represents both unextractable and missing data, we did not have a way of classifying documents and we did not extract any family relationship.
- **3. Do [i] [know] a [plumber]? :** We did not identified anyones occupation even tho there where signatures in emails containing this information. The titles where ignored or identified as named entities. Location data and the information obtained on businesses contain disambiguation information. This allows us to identify what categories organizations falls in under and answer queries like ’What plumber did we use last time?’ and ’Call the plumber we used last time?’.

5.4.2 Not covered by the ontology

Some queries where not possible to answer because we did not have the structure in place to answer them, these examples are queries dependent on external resources to answer.

- **1. Are there any interesting concerts next week?** Our PKG contain artist and genres of interest to the user along with global identifiers, relevant resources and URI of artist in public knowledge graphs. Answering this would require using API of external resources like Songkick [68], which we did not.
- **2. Where can i get a [drive belt] for [my scooter]?** We do not have integration with a service that would allow us to answer this.

Chapter 6

Discussion

In this chapter we will discuss our findings from our work towards building a Personal Knowledge Graph.

In section 6.1 we will talk about the ontology used, its viability as a foundation for a PKG and its strength and weaknesses. In Section 6.3 we discuss our approach and the challenges we encountered collecting data, extracting and building our knowledge graph. Section 6.4 we discuss the practical usability of the PKG.

6.1 Ontology

We wanted to know if existing ontologies could fulfil the requirements defined by our competency questions, if existing could be adapted or merged or if it was necessary to build our own from scratch. The final rendition of our ontology included the FOAF, RELATIONSHIP, PersonLink, schema.org and a few proprietary classes.

The use of general purpose vocabularies and domain specific ontologies worked well, however we did encounter implementation issues with the PersonLink ontology, and the combination of these ontologies alone did not cover all the use cases. This was private information not applicable for KGs with more then one user, like opinions.

The markup vocabulary schema.org does not provide description logic to its vast number of classes, meaning no complex reasoning and inference. The size and popularity of this vocabulary is a strong advantage, as information embedded in emails, on web pages and responses from proprietary APIs using this vocabulary can be integrated to the PKG with little effort, negating the need for interpretation and restructuring. Examples of this are orders, reservations or bookings, found embedded in emails, that could be extracted and included to the PKG directly, leaving only the tasks of entity linking and data transformation.

The RELATIONSHIP vocabulary is an existing extension of the FOAF-ontology and define detailed relationships useful for the PKG. It does also describe family relationships including logical descriptors, however we wanted a more expressive model for family relationships. The PersonLink ontology is able to model complex family relationships, it is adaptable to cultural differences that may occur when describing family relationships and it is multilingual. Because there is no universally supported syntax for expressing these rules among triplestores and ontology editors, we were not able to integrate PersonLink fully. The syntax used to define the logic rules in PersonLink is not directly supported by the triplestore we used and the share number of rules made it too large a project to translate. PersonLink was kept in our final PKG ontology because we believe it has a practical value in a PKG, even though we were not able to utilize its full potential in our graph.

The inclusion of vCard object-properties into the ontology worked well. It made conversion from vCard to RDF trivial, allowing us easily to include more detailed properties than the FOAF-profile has, using a known vocabulary. We encountered one small problem with this method, due to the input data. The object property `foaf:firsName` equivalent to `vcard:given-name` has the domain `foaf:Person`, this resulted in companies stored with the name in the wrong field in the original vCard being inferred as a `foaf:Person` in the PKG.

Experimentation with adding labelled relationships depicting how person A may refer to person B, proved useful when simplifying queries from the user's perspective. The method is depending on unique rdf:labels for object properties and therefor a bit risky.

6.1.1 Provenance

The uncertainty associated with knowledge extraction using techniques with low precision and the short-lived nature of some of the information contained within the PKG, makes it necessary to annotate triples with metadata like confidence, source and timestamp. RDF-STAR was used to include provenance data because it is less cumbersome to implement, and results in less complicated queries when using this technique compared to traditional rectification. However, while the method will let you annotate a specific relationship with metadata, it will not provide a way to uniquely distinguish triples, for this we need named graphs or rectification. Another issue we had with our implementation of RDF-star was that the OWL reasoner did not do inference on embedded triples.

One advantage of RDF-Star dealing with uncertain data extracted from text, is that RDF-STAR triples can easily be included into the graph without asserting the triples it describe. This allows us to manage noise, by asserting triples based on confirmation from multiple text in addition to the estimated confidence value.

6.1.2 Evaluating the ontology

The resulting *Knowledge Graph* built on the *ontology* described in this thesis is capable of answering most of the competency question defined, but missing queries dependent on knowledge from third party providers. However the ontology is built using a small set of questions, selected by us, restricting the evaluation to a demonstration of its potential, rather than revealing limitations. Building a collection of competency questions from a larger group of subjects would have given a better foundation for evaluation and quite possibly revealed use cases that are not covered by our ontology.

The PKG may be centered around one entity, the user, but the domain of the graph is still large, due to the fact that anything and anyone of interest to the user is considered useful for the graph. Based on a small set of competency questions to be answered, the structure could be modelled numerous ways and still be able to answer the questions. We cannot conclude that our ontology is the ideal way of modelling a PKG, but it is an functional, practical and easily expandable ontology.

6.2 Information Extraction on informal text

In this section we will discuss the challenges we encountered with information extraction on informal text. Firstly in Section 6.2.1 the challenges with obtaining data for IE on informal text and secondly in Section 6.2.2 the methods used.

6.2.1 Datasets for Information Extraction

With sparse availability of datasets containing personal correspondence suitable for training and evaluating *Information Extraction* on informal text, most research have focused on public text sources like blog posts and comments [32, 36, 37] [39] or using proxies like movie scripts [36]. When the perspective is developing and evaluating IE models on informal text, these types of text are within the domain and appropriate to use. Using these types of text as proxies for personal correspondence and for the purpose of building a PKG however, might not be appropriate. Our decision to use genuine correspondence for IE however, did reduce possible datasets available and made the task a lot more challenging.

The biggest challenge we faced trying to obtain personal knowledge from correspondence is the sparsity of this type of information and the lack of context. Our experimentation with public-post, the Enron-set and our own personal correspondence revealed that there is a significant difference in provided context and language between informal text written for an unknown public and text intended for specific recipients.

Accessing enough genuine personal correspondence to train NLP models for this specific purpose might be difficult. These texts are highly variant, depending on personality, dialect and culture in addition to the language they are written in. Using a small dataset as a foundation and techniques for generating synthetic data may therefore lead to a biased dataset.

6.2.2 Information Extraction methods

Our approach using publicly available pretrained models developed on large collections of public web-data as a foundation to extract structured knowledge from correspondence yielded no applicable results.

The models used where all pipeline-models, where downstream tasks are fully or partially dependent prior steps. This quickly leads to error propagation, where mistakes early on leads to subsequent tasks to fail. Apart from misspellings and badly structured sentences there where a key factor leading to failure that stood out during our testing. This was missing or misplaced punctuation, causing errors in sentence segmentation resulting in dependency parsing, noun chunking and subsequent relation extraction to fail for two sentences. The number of documents containing relevant relationships where sparse and far from enough to train a new model from scratch or a new model based on features identified by the pretrained pipeline models. This left us with the naive approach using hand built patterns detecting relationships based on POS-tagging, dependency parsing and NER. The high variance in sentence structures and the amount of errors made by the pretrained models made the task of relation extraction using patterns unsuccessful.

Models using a gazetteer and pattern recognition in the pipeline had significant advantages when the sentence structures where hard to interpret. This types of methods is however limited to entity names already in the system.

The short text and the sparsity of relevant examples where we would be dependent co-reference resolution, meant we were not able to evaluate these models. Persons in personal correspondence is however often referenced by personal pronouns, which is easily resolvable, knowing the author and the recipient(s).

6.3 Building the knowledge graph

To construct our graph we needed to obtain relevant data from personal sources, extract structured knowledge and restructure these to fit our ontology. In this section we discuss the process and result from building our own PKG using personal data sources.

How much, what type and in what form personal knowledge is contained within a user's resources will naturally vary from person to person and we can only base our finding on a limited dataset from our own sources.

6.3.1 NOR emails - processing and Information Extraction

The use of a dataset collected from one user, do lead to the possibility of identifying and linking entities cross multiple disparate sources, creating one coherent knowledge structure. However, the use of a relatively small dataset from only one user come with some major caveats when evaluating the methods for information extraction. The number of correspondents in the email-set is way too small to create an unbiased test-set.

Prepossessing

The methods used to filter out and ignore irrelevant emails, did have usable result, to some extent. It did not remove important emails or any informative text, a lot of irrelevant emails were however, not filtered out and a lot of noisy text was fed to the IE pipeline.

The method to identify signatures with contact information did not have usable result, building patterns or training models to do this with good accuracy is possible for example by adapting the methods used by mailgun [63]. We did not explore this task further.

We were not able to classify the entity type of the email senders. Names and domains will make it possible to do so to some extent using external services or dictionaries. Distinguishing between correspondence between the user and private persons and between the user and organizations is of value to the PKG. Emails from unknown entities was classified as `foaf:Agent` same as in [42]. Only existing entities populated from location data and contact information distinguished between Persons and Organizations.

Disambiguation

The small dataset meant that we did not have any genuine conflicts disambiguation records against the PKG. The normal challenges with ambiguity did not occur.

Information Extraction

The results of our IE model on our private emails did not result in anything more than recognising entity mentions and contact information by using the entity ruler and simple pattern of phone numbers, emails and dates. The pretrained NER-model of the Norwegian pipeline generated a lot more false positives than detecting names and classifying them.

While identifying mentions in informal text prove challenging for models depending on POS-tagging and dependency parsing, due to the nature of informal sentence structures, the user's PKG includes a relatively small set of named entities relevant to the user. The names and nicknames of these entities can easily be used to identify mentions, classify them and in some cases disambiguate entities directly, by adding an entity ruler(dictionary) to the pipeline. For NER on informal text with 'creative' sentence-structures, this type of approach might be necessary, although it is dependent on the entity to be referenced by a known name or nick. The method did work well, however limited to exact matches as implemented.

We did not enrich our PKG with entity mentions and property values found in our emails. However, adding this type of information to the PKG will potentially add some practical value. Making it possible to answer question like 'Give me the directions to the address my mother sent me.'.

Embedded metadata in Emails

Providers like *Google* are encouraging organizations to embed metadata using the schema.org vocab into their automated responses so this information can be highlighted and used in their services like search, notifications and DPAs.

However, we do not have data on how common it is to use embedded metadata in email, nor do we know if this is an upcoming or dying trend. We know that this technique along with the use of known vocabularies allows applications to automatically detect and utilize information like order confirmations and shipping information. The viability of the PKG concept would improve greatly if the practice of including easily accessible, relevant and structured data in emails were to become common practice.

6.3.2 Potential sources for a PKG

This project included only a small portion of potential data sources available to a user. Social networks, messaging apps, cloud services and mobile applications are potential sources containing structured, semi-structured and unstructured knowledge relevant for a PKG.

We used the data from a cloud service provider to obtain location data, otherwise including location data in any practical usable way would be a project of its own, however there are a lot of potentially valuable personal data left alone from this provider alone.

6.3.3 The system-framework

Our intention was to conceptualise a framework capable of continuously obtaining/synchronizing data from multiple disparate sources, running the appropriate processing on the data and populate the PKG. A concept of a framework was developed, but important details were not investigated, like security and synchronisation. The personal data populating our PKG where for the most part static files that we extract, process and populate our graph with, running the process as a simple script.

This project has not dived deep into the issues and complications that may occur when collecting data from multiple sources. However, it is apparent that uniformly adapted standards and common vocabularies simplifies this task significantly where they exist. The vast number of APIs and proprietary vocabularies that would need to be utilized to create an application capable of including data from all common resources makes it unfeasible.

The methods used to process, restructure and populate the PKG worked well and made the process of including additional structured data sources to the PKG trivial, given that the entity types are covered by the ontology. OTTR-templates were only used for static location data because of the very basic interface used in our implementation to interact with the tool to expand triples, Lutra. A more seamless integration using these templates would potentially facilitate integration of new entity types and new data sources in a more streamlined manner.

6.4 Using the Personal Knowledge Graph

We wanted to know how the personal knowledge graph can be used, its advantages and disadvantages over comparable solutions. Our interface with the PKG in this project was dependent on SPARQL, either programmatically or by using the Fuseki servers UI. This does not represent the intended user interface for the PKG, as it is meant as a resource for systems like DPAs.

The competency questions defining the scope of the ontology and the use cases it can accommodate are selected by us and therefore somewhat biased and limited by our own imagination. The questions were formulated with practical use in mind and include several common use cases for DPAs. This way of defining and selecting competency questions did result in large PKG with a large number of use cases, but it does not emphasise the possibilities that are unique for a PKG, nor does it reveal the limitations of the PKG.

Balog and Kenter:[2] includes examples of short lived information like “the ingredients of the dinner I planned on cooking tonight” in their vision of a PKG. Including and querying this type of short lived information was not explored. While it is possible to represent the knowledge in this example in the PKG, the ontology as is, makes it impractical to utilise the information.

The PKG built solely on knowledge extracted from personal sources closely resembled a graph structured PIMS, a coherent source of all the user's resources like emails, contact information and calendar entries. However, by enriching the PKG with some key information like personal relationships and nicknames, the ease of interacting and obtaining these resources was improved significantly. The possibility of asking questions like 'Get me the document my mother sent me', without specifying the service it was sent to (emails, chat, ++) or the address of the sender, is practical and show the value of aggregating user data into one source.

Research focused on PKGs frequently mentions the growing popularity of Digital Personal Assistants as the motivation behind building such a graph [2]. Current DPAs are able to store these types of personal relationships and to some extent do inferring based on this information, helpful for user queries. However, these services are limited to their own ecosystem, potentially limiting the available data to only a small portion of the user's data.

The PKG allows the user to have control and awareness over how their personal knowledge is modeled, one of the downsides with current DPAs. The users willingness to share their personal interaction and data with the service providers DPAs is however a valuable resource for developing and improving these services.

The way service providers of DPAs develop their models and finance their services by using data provided by their customers has an impact in how users interact with their DPAs. The Voice Report [23] shows that consumers are concerned about the privacy of their personal information to the extent that they are reluctant to divulge it to a DPA. This, along with the fact that DPAs are restricted to the providers ecosystem limits the potential use of the DPA.

Chapter 7

Conclusion and Future Directions

The goal of this project was to conceptualize a framework for aggregating user data into one uniform source of personal information and to explore the viability and usability of a Personal Knowledge Graph automatically created from the user's resources.

We have shown that a PKG based on the user's personal resources is a viable concept. We have proposed a framework for aggregating user data and shown how existing ontologies can be used to model personal knowledge. We shown how this knowledge can be leveraged by a digital personal assistant and how it can be used in the context of personal information management.

However, there are challenges to be solved before practical applications based on this framework may be developed.

Extracting structured knowledge from unstructured personal sources is by a good margin the most challenging task among them.

The main challenge we encountered with IE on personal sources was sparsity of relevant information and the lack of context. The informal language also proved difficult for the pre-trained models to interpret. However, personal correspondence and the domain defined by our PKG is outside the scope these models where trained for.

Development of purpose built models trained on appropriate document collection with the domain defined by the PKG in mind, might be necessary to achieve the desired level of extraction.

There are also challenges with data synchronization, security, systems development and creating user interfaces with high usability. These require comprehensive work, but are technically feasible using available methods, as PIMS like THYMEFLOW [42] shows.

The resulting use cases covered by our ontology and answerable by the automatically populated PKG, did not extend beyond the capabilities of modern DPAs. However, it is capable of consolidating data from multiple disparate sources and do not confine the user to one ecosystem. It also gives the user insight and control over how their knowledge is modelled and stored. Lack of understanding, insight and control over personal data are aspects holding users back from utilizing the full potential of modern DPAs.

The research and efforts towards data standardisation and personal ownership of personal data, like the solid project[69], would be a major contributor in realizing the vision of a secure, personal digital assistant based on a personal knowledge graph using personal data from multiple sources.

List of Figures

List of Tables

Bibliography

[1] Mohammad Nasar, Masnizah Mohd, and Nazlena Mohamad Ali. Personal information management systems and interfaces: An overview. pages 197 – 202, 07 2011. doi: 10.1109/STAIR.2011.5995788.

[2] Krisztian Balog and Tom Kenter. Personal knowledge graphs: A research agenda. In *Proceedings of the ACM SIGIR International Conference on the Theory of Information Retrieval (ICTIR)*, 2019.

[3] Martin Swain. *Knowledge Representation*, pages 1082–1084. Springer New York, New York, NY, 2013. ISBN 978-1-4419-9863-7. doi: 10.1007/978-1-4419-9863-7_595. URL https://doi.org/10.1007/978-1-4419-9863-7_595.

[4] Krisztian Balog. *Entity-Oriented Search*. 10 2018. ISBN 978-3-319-93935-3. doi: 10.1007/978-3-319-93935-3.

[5] URL http://www.cs.man.ac.uk/~stevensr/onto/node14.html.

[6] Phillip Lord. Components of an ontology, Jan 2010. URL http://ontogenesis.knowledgeblog.org/514/.

[7] Tom Gruber. URL http://tomgruber.org/writing/ontology-definition-2007.htm.

[8] R. Riet and R. Meersman. Linguistic instruments in knowledge engineering. 1992.

[9] Lisa Ehrlinger and Wolfram Wöß. Towards a definition of knowledge graphs. 09 2016.

[10] . URL https://www.w3.org/TR/rdf11-concepts/.

[11] URL https://www.w3.org/TR/n-quads/.

[12] . URL https://w3c.github.io/rdf-star/cg-spec.

[13] Olaf Hartig. Foundations of rdf and sparql (an alternative approach to statement-level metadata in rdf). 06 2017.

[14] Filipe Mesquita, Matteo Cannaviccio, Jordan Schmidek, Paramita Mirza, and Denilson Barbosa. KnowledgeNet: A benchmark dataset for knowledge base population. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 749–758, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1069. URL https://www.aclweb.org/anthology/D19-1069.

[15] N. Gao, Mark Dredze, and Douglas W. Oard. Knowledge base population for organization mentions in email. In *AKBC@NAACL-HLT*, 2016.

[16] Alan Ritter, Sam Clark, Mausam, and Oren Etzioni. Named entity recognition in tweets: An experimental study. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1524–1534, Edinburgh, Scotland, UK., July 2011. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D11-1141.

[17] D. Etter, F. Ferraro, Ryan Cotterell, Olivia Buzek, and Benjamin Van Durme. Nerit: Named entity recognition for informal text. 2013.

[18] Einat Minkov, Richard Wang, and William Cohen. Extracting personal names from email: Applying named entity recognition to informal text. 01 2005.

[19] Lingraj S Vannur, Balaji Ganesan, Lokesh Nagalapatti, Hima Patel, and MN Thippeswamy. Data augmentation for personal knowledge base population, 2020.

[20] Ofer Bergman, Richard Boardman, Jacek Gwizdka, and William Jones. Personal information management. 03 2004. doi: 10.1145/985921.986164.

[21] Tudor Groza, Siegfried Handschuh, Knud Möller, Gunnar Grimnes, Leo Sauermann, Enrico Minack, Cédric Mesnage, Mehdi Jazayeri, Gerald Reif, and Rosa Gudjonsdottir. The nepomuk project—on the way to the social semantic desktop. pages 201–211, 06 2010.

[22] Adam Cheyer, J. Park, and Richard Giuli. Iris: Integrate. relate. infer. share. In *Semantic Desktop Workshop*, 2005.

[23] Christi OlsonSearch Evangelist; Microsoft Search, Christi Olson, Search Evangelist; Microsoft Search, Kelli KemeryResearcher; Microsoft Market Intelligence, Kelli Kemery, and Researcher; Microsoft Market Intelligence. 2019 microsoft voice report, Dec 2020. URL https://about.ads.microsoft.com/en-us/insights/2019-voice-report.

[24] Ewa Luger and Abigail Sellen. "like having a really bad pa": The gulf between user expectation and experience of conversational agents. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, page 5286–5297, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450333627. doi: 10.1145/2858036.2858288. URL `https://doi.org/10.1145/2858036.2858288`.

[25] Anna Tigunova. Extracting personal information from conversations. *Companion Proceedings of the Web Conference 2020*, 2020.

[26] Anna Tigunova, Paramita Mirza, Andrew Yates, and Gerhard Weikum. Exploring personal knowledge extraction from conversations with charm. pages 1077–1080, 03 2021. doi: 10.1145/3437963.3441699.

[27] D. Montoya, Thomas Pellissier Tanon, S. Abiteboul, P. Senellart, and Fabian M. Suchanek. A knowledge base for personal information management. In *LDOW@WWW*, 2018.

[28] Foaf vocabulary specification 0.99. URL `http://xmlns.com/foaf/spec/20140114.html`.

[29] Eleni Kargioti, Efstratios Kontopoulos, and Nick Bassiliades. Ontolife: an ontology for semantically managing personal information. In Iliadis, Maglogiann, Tsoumakasis, Vlahavas, and Bramer, editors, *Artificial Intelligence Applications and Innovations III*, pages 127–133, Boston, MA, 2009. Springer US. ISBN 978-1-4419-0221-4.

[30] Tara Safavi, Adam Fourney, Robert Sim, Marcin Juraszek, Shane Williams, Ned Friend, Danai Koutra, and Paul N. Bennett. Toward activity discovery in the personal web. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, WSDM '20, page 492–500, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450368223. doi: 10.1145/3336191.3371828. URL `https://doi.org/10.1145/3336191.3371828`.

[31] Keith Cortis and C. Abela. Semchat: Extracting personal information from chat conversations. 2010.

[32] Anna Tigunova, Andrew Yates, Paramita Mirza, and Gerhard Weikum. CHARM: Inferring personal attributes from conversations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5391–5404, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.434. URL `https://www.aclweb.org/anthology/2020.emnlp-main.434`.

[33] V. Agarwal, O. Z. Khan, and R. Sarikaya. Remembering what you said: Semantic personalized memory for personal digital assistants. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5835–5839, 2017. doi: 10.1109/ICASSP.2017.7953275.

[34] Liangchen Luo, Wenhao Huang, Qi Zeng, Zaiqing Nie, and Xu Sun. Learning personalized end-to-end goal-oriented dialog. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):6794–6801, Jul. 2019. doi: 10.1609/aaai.v33i01.33016794. URL https://ojs.aaai.org/index.php/AAAI/article/view/4654.

[35] Yuanchun Li, Ziyue Yang, Yao Guo, Xiangqun Chen, Yuvraj Agarwal, and Jason I. Hong. Automated extraction of personal knowledge from smartphone push notifications. *CoRR*, abs/1808.02013, 2018. URL http://arxiv.org/abs/1808.02013.

[36] Anna Tigunova, Andrew Yates, Paramita Mirza, and Gerhard Weikum. Listening between the lines: Learning personal attributes from conversations, 2019.

[37] Anna Tigunova. Extracting personal information from conversations. *Companion Proceedings of the Web Conference 2020*, 2020.

[38] Hongyan Jing, Nanda Kambhatla, and Salim Roukos. Extracting social networks and biographical facts from conversational speech transcripts. 01 2007.

[39] An-Zi Yen, Hen-Hsen Huang, and H. Chen. Personal knowledge base construction from text-based lifelogs. *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2019.

[40] X. Li, G. Tur, D. Hakkani-Tür, and Q. Li. Personal knowledge graph population from user utterances in conversational understanding. In *2014 IEEE Spoken Language Technology Workshop (SLT)*, pages 224–229, 2014. doi: 10.1109/SLT.2014.7078578.

[41] D. Montoya, Thomas Pellissier Tanon, S. Abiteboul, P. Senellart, and Fabian M. Suchanek. Thymeflow, an open-source personal knowledge base system. 2016.

[42] David Montoya, Thomas Tanon, Serge Abiteboul, and Fabian Suchanek. Thymeflow, a personal knowledge base with spatio-temporal data. pages 2477–2480, 10 2016. doi: 10.1145/2983323.2983337.

[43] D. Crocker. Rfc0822: Standard for the format of arpa internet text messages. 1982.

[44] Bernard Desruisseaux. Internet calendaring and scheduling core object specification (icalendar). 09 2009.

[45] Ludovic Langrand. Clean your inbox, plant trees! URL https://www.cleanfox.io/en/.

[46] Onmail – ultra-fast, dead simple email by edison. URL https://mail.edison.tech/.

[47] Daniel Preoţiuc-Pietro, Vasileios Lampos, and Nikolaos Aletras. An analysis of the user occupational class through Twitter content. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1754–1764, Beijing, China, July 2015. Association for Computational Linguistics. doi: 10.3115/v1/P15-1169. URL https://www.aclweb.org/anthology/P15-1169.

[48] Martin G. Skjæveland, . URL https://ottr.xyz/#Specifications.

[49] Mark A. Musen. The protégé project: a look back and a look forward. *AI Matters*, 1(4):4–12, 2015. doi: 10.1145/2757001.2757003. URL https://doi.org/10.1145/2757001.2757003.

[50] Steffen Lohmann, Vincent Link, Eduard Marbach, and Stefan Negru. Webvowl: Web-based visualization of ontologies. volume 8982, pages 154–158, 04 2015. ISBN 978-3-319-17965-0. doi: 10.1007/978-3-319-17966-7_21.

[51] Reasoners and rule engines: Jena inference support. URL https://jena.apache.org/documentation/inference/.

[52] Martin G. Skjæveland, . URL https://spec.ottr.xyz/tabOTTR/0.3/.

[53] Serge Abiteboul, Benjamin André, and Daniel Kaplan. Managing your digital life with a personal information management system. 05 2015.

[54] Michael Grüninger and Mark Fox. Methodology for the design and evaluation of ontologies. 07 1995.

[55] Yuan Ren, Artemis Parvizi, Chris Mellish, Jeff Z. Pan, Kees van Deemter, and Robert Stevens. Towards competency question-driven ontology authoring. In Valentina Presutti, Claudia d'Amato, Fabien Gandon, Mathieu d'Aquin, Steffen Staab, and Anna Tordai, editors, *The Semantic Web: Trends and Challenges*, pages 752–767, Cham, 2014. Springer International Publishing. ISBN 978-3-319-07443-6.

[56] Noura Herradi, Fayçal Hamdi, Elisabeth Métais, Fatma Ghorbel, and Assia Soukane. Personlink: An ontology representing family relationships for the captain memo memory prosthesis. pages 3–13, 10 2015. ISBN 978-3-319-25746-4. doi: 10.1007/978-3-319-25747-1_1.

[57] Harry Halpin. URL http://www.ibiblio.org/hhalpin/homepage/notes/vcardtable.html.

[58] visited. URL https://www.thefreedictionary.com/visited.

[59] Opinion: Definition of opinion by oxford dictionary on lexico.com also meaning of opinion. URL https://www.lexico.com/definition/Opinion.

[60] Provenance, authoring and versioning. URL https://pav-ontology.github.io/pav/.

[61] recordlinkage. URL https://pypi.org/project/recordlinkage/.

[62] Beautiful soup documentation¶. URL https://www.crummy.com/software/BeautifulSoup/bs4/doc/#.

[63] Mailgun. mailgun/talon. URL https://github.com/mailgun/talon.

[64] Norwegian bokmål · spacy models documentation. URL https://spacy.io/models/nb.

[65] URL https://www.cs.cmu.edu/~enron/.

[66] Einat Minkov, Richard C. Wang, and William W. Cohen. Extracting personal names from email: Applying named entity recognition to informal text. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 443–450, Vancouver, British Columbia, Canada, October 2005. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/H05-1056.

[67] Ning Gao, Mark Dredze, and Douglas Oard. Person entity linking in email with nil detection. *Journal of the Association for Information Science and Technology*, 68, 07 2017. doi: 10.1002/asi.23888.

[68] Find your perfect concert wherever you are. URL https://www.songkick.com/.

[69] URL https://solidproject.org/.