CREATION, ENRICHMENT AND APPLICATION OF KNOWLEDGE GRAPHS

Von der Fakultät für Elektrotechnik und Informatik
der Gottfried Wilhelm Leibniz Universität Hannover
zur Erlangung des Grades

DOKTOR DER NATURWISSENSCHAFTEN

Dr. rer. nat.

genehmigte Dissertation
von

M. Sc. Simon Gottschalk

geboren am 22. April 1992 in Hannover, Deutschland

Hannover, Deutschland, 2021

Referent: Prof. Dr. techn. Wolfgang Nejdl
Korreferent: Prof. Dr. Elena Demidova
Korreferent: Prof. Dr. Sören Auer
Tag der Promotion: 29. April 2021

ABSTRACT

The world is in constant change, and so is the knowledge about it. Knowledge-based systems – for example, online encyclopedias, search engines and virtual assistants – are thus faced with the constant challenge of collecting this knowledge and beyond that, to understand it and make it accessible to their users. Only if a knowledge-based system is capable of this understanding – that is, it is capable of more than just reading a collection of words and numbers without grasping their semantics – it can recognise relevant information and make it understandable to its users. The dynamics of the world play a unique role in this context: Events of various kinds which are relevant to different communities are shaping the world, with examples ranging from the coronavirus pandemic to the matches of a local football team. Vital questions arise when dealing with such events: How to decide which events are relevant, and for whom? How to model these events, to make them understood by knowledge-based systems? How is the acquired knowledge returned to the users of these systems?

A well-established concept for making knowledge understandable by knowledge-based systems are *knowledge graphs*, which contain facts about entities (persons, objects, locations, ...) in the form of graphs, represent relationships between these entities and make the facts understandable by means of ontologies. This thesis considers knowledge graphs from three different perspectives: (i) *Creation of knowledge graphs*: Even though the Web offers a multitude of sources that provide knowledge about the events in the world, the creation of an event-centric knowledge graph requires recognition of such knowledge, its integration across sources and its representation. (ii) *Knowledge graph enrichment*: Knowledge of the world seems to be infinite, and it seems impossible to grasp it entirely at any time. Therefore, methods that autonomously infer new knowledge and enrich the knowledge graphs are of particular interest. (iii) *Knowledge graph interaction*: Even having all knowledge of the world available does not have any value in itself; in fact, there is a need to make it accessible to humans. Based on knowledge graphs, systems can provide their knowledge with their users, even without demanding any conceptual understanding of knowledge graphs from them. For this to succeed, means for interaction with the knowledge are required, hiding the knowledge graph below the surface.

In concrete terms, I present *EventKG* – a knowledge graph that represents the happenings in the world in 15 languages – as well as *Tab2KG* – a method for understanding tabular data and transforming it into a knowledge graph. For the enrichment of knowledge graphs without any background knowledge, I propose *HapPenIng*, which infers missing events from the descriptions of related events. I demonstrate means for interaction with knowledge graphs at the example of two web-based systems (*EventKG+TL* and *EventKG+BT*) that enable users to explore the happenings in the world as well as the most relevant events in the lives of well-known personalities.

Key words *knowledge graphs, events, Semantic Web, creation of knowledge graphs, enrichment of knowledge graphs, interaction with knowledge graphs*

ZUSAMMENFASSUNG

Die Welt befindet sich im steten Wandel, und mit ihr das Wissen über die Welt. Wissensbasierte Systeme – seien es Online-Enzyklopädien, Suchmaschinen oder Sprachassistenten – stehen somit vor der konstanten Herausforderung, dieses Wissen zu sammeln und darüber hinaus zu verstehen, um es so Menschen verfügbar zu machen. Nur wenn ein wissensbasiertes System in der Lage ist, dieses Verständnis aufzubringen – also zu mehr in der Lage ist, als auf eine unsortierte Ansammlung von Wörtern und Zahlen zurückzugreifen, ohne deren Bedeutung zu erkennen –, kann es relevante Informationen erkennen und diese seinen Nutzern verständlich machen. Eine besondere Rolle spielt hierbei die Dynamik der Welt, die von Ereignissen unterschiedlichster Art geformt wird, die für unterschiedlichste Bevölkerungsgruppe relevant sind; Beispiele hierfür erstrecken sich von der Corona-Pandemie bis hin zu den Spielen lokaler Fußballvereine. Doch stellen sich hierbei bedeutende Fragen: Wie wird die Entscheidung getroffen, ob und für wen derlei Ereignisse relevant sind? Wie sind diese Ereignisse zu modellieren, um von wissensbasierten Systemen verstanden zu werden? Wie wird das angeeignete Wissen an die Nutzer dieser Systeme zurückgegeben?

Ein bewährtes Konzept, um wissensbasierten Systemen das Wissen verständlich zu machen, sind *Wissensgraphen*, die Fakten über Entitäten (Personen, Objekte, Orte, ...) in der Form von Graphen sammeln, Zusammenhänge zwischen diesen Entitäten darstellen, und darüber hinaus anhand von Ontologien verständlich machen. Diese Arbeit widmet sich der Betrachtung von Wissensgraphen aus drei aufeinander aufbauenden Blickwinkeln: (i) *Erstellung von Wissensgraphen*: Auch wenn das Internet eine Vielzahl an Quellen anbietet, die Wissen über Ereignisse in der Welt bereithalten, so erfordert die Erstellung eines ereigniszentrierten Wissensgraphen, dieses Wissen zu erkennen, miteinander zu verbinden und zu repräsentieren. (ii) *Anreicherung von Wissensgraphen*: Das Wissen über die Welt scheint schier unendlich und so scheint es unmöglich, dieses je vollständig (be)greifen zu können. Von Interesse sind also Methoden, die selbstständig das vorhandene Wissen erweitern. (iii) *Interaktion mit Wissensgraphen*: Selbst alles Wissen der Welt bereitzuhalten, hat noch keinen Wert in sich selbst, vielmehr muss dieses Wissen Menschen verfügbar gemacht werden. Basierend auf Wissensgraphen, können wissensbasierte Systeme Nutzern ihr Wissen darlegen, auch ohne von diesen ein konzeptuelles Verständis von Wissensgraphen abzuverlangen. Damit dies gelingt, sind Möglichkeiten der Interaktion mit dem gebotenen Wissen vonnöten, die den genutzten Wissensgraphen unter der Oberfläche verstecken.

Konkret präsentiere ich *EventKG* – einen Wissensgraphen, der Ereignisse in der Welt repräsentiert und in 15 Sprachen verfügbar macht, sowie *Tab2KG* – eine Methode, um in Tabellen enthaltene Daten anhand von Hintergrundwissen zu verstehen und in Wissensgraphen zu wandeln. Zur Anreicherung von Wissensgraphen ohne weiteres Hintergrundwissen stelle ich *HapPenIng* vor, das fehlende Ereignisse aus den vorliegenden Beschreibungen ähnlicher Ereignisse inferiert. Interaktionsmöglichkeiten mit Wissensgraphen demonstriere ich anhand zweier web-basierter Systeme (*EventKG+TL* und *EventKG+BT*), die Nutzern auf einfache Weise die Exploration von Geschehnissen in der Welt sowie der wichtigsten Ereignisse in den Leben bekannter Persönlichkeiten ermöglichen.

Schlagwörter: *Wissensgraphen, Ereignisse, Semantic Web, Erstellung von Wissensgraphen, Anreicherung von Wissensgraphen, Interaktion mit Wissensgraphen*

ACKNOWLEDGMENTS

First of all, I would like to thank Prof. Dr. techn. Wolfgang Nejdl, who has supervised my path as a PhD student from the first day on and thus made it possible for me to write up this thesis finally. Second, I would like to thank Prof. Dr. Elena Demidova, who accompanied me closely over the past years at the L3S, which in the end has not only lead to a bunch of joint publications but also to an uncountable amount of inspiring discussions. Third, my thanks go to Prof. Dr. Sören Auer for his time and effort to examine this thesis.

With more than five years at the L3S Research Center, I have seen many colleagues come and go. With many of them, I had not only intense scientific discussions but also shared great moments outside the research institutions, and it is not without difficulties to mention all of them here. Without a doubt, I have spent the most time with my office mates Nicolas and Markus. This time includes all our conversations whenever we needed to have a break and look up from our screens. The office right next to ours was occupied by Renato and Tarcísio most of the time. They had started at L3S just briefly before me and thus accompanied and enriched a large part of my time as a PhD student. Apart from these two offices at the 15th floor of the L3S, there have been the "KBS guys" Besnik, Bill, Christoph, Ujwal as well as many others, including but not limited to Asmelash, Damianos, Jaspreet, Jurek, Lijun, Mary, Ran and Sergej. More recently, I started collaborations with Sara, Tin and Fakhar. I would also like to thank the colleagues working in the administrative and technical departments, such as Dimitar, Miroslav and others.

Outside of the L3S, at conferences, during research visits and via project meetings, I have met many great researchers, including Paolo and several others. Even if we only met for a few days, we will always have some common memories to share.

Finally, I would like to thank my family, in particular my parents, sister and girlfriend, as well as my friends. They had accompanied my path as a PhD student, endured my non-availability when it came close to submission deadlines, and never stopped providing me with valuable pieces of advice.

The works presented in this thesis were partially funded by the ERC under ALEXANDRIA (339233), by the Federal Ministry of Education and Research (BMBF), Germany, under Simple-ML (01IS18054) and by DFG, German Research Foundation, under WorldKG (DE 2299/2-1).

FOREWORD

The work presented in this thesis has been published at various conferences and journals, as follows.

Chapter 3 is built on the following works about my knowledge graph *EventKG*:

- Simon Gottschalk and Elena Demidova. EventKG: A Multilingual Event-centric Temporal Knowledge Graph. Extended Semantic Web Conference (ESWC), 2018. (Resource paper) [GD18a]
- Simon Gottschalk and Elena Demidova. EventKG - the Hub of Event Knowledge on the Web - and Biographical Timeline Generation. Semantic Web Journal (SWJ), 2019. (Full Journal paper) [GD19a]

In Chapter 4, I describe the following work:

- Simon Gottschalk and Elena Demidova. Tab2KG: Transforming Tabular Data into Knowledge Graphs using Semantic Domain Profiles. (Under submission)

Chapter 5 presents the research published in:

- Simon Gottschalk and Elena Demidova. HapPenIng: Happen, Predict, Infer — Event Series Completion in a Knowledge Graph. International Semantic Web Conference (ISWC), 2019. (Full paper) [GD19b]

Chapter 6 again builds up on the *EventKG* Journal paper, as well as two demonstration papers:

- Simon Gottschalk and Elena Demidova. EventKG - the Hub of Event Knowledge on the Web - and Biographical Timeline Generation. Semantic Web Journal (SWJ), 2019. (Full Journal paper) [GD19a]
- Simon Gottschalk and Elena Demidova. EventKG+BT: Generation of Interactive Biography Timelines from a Knowledge Graph. Extended Semantic Web Conference (ESWC), 2020. (Demonstration paper) [GD20]

- Simon Gottschalk and Elena Demidova. EventKG+TL: Creating Cross-Lingual Timelines from an Event-Centric Knowledge Graph. Extended Semantic Web Conference (ESWC), 2018. (Demonstration paper) [GD18b]

The complete list of publications during my PhD follows:

Journal articles

- Simon Gottschalk and Elena Demidova. EventKG - the Hub of Event Knowledge on the Web - and Biographical Timeline Generation. Semantic Web Journal (SWJ), 2019. (Full Journal paper) [GD19a]
- Simon Gottschalk and Elena Demidova. MultiWiki: Interlingual Text Passage Alignment in Wikipedia. ACM Transactions on the Web (TWEB), 2018. (Full Journal paper) [GD17]

Conference papers

- Tarcísio Souza Costa, Simon Gottschalk and Elena Demidova. Event-QA: A Dataset for Event-Centric Question Answering over Knowledge Graphs. Conference on Information and Knowledge Management (CIKM), 2020. (Resource paper) [CGD20]
- Simon Gottschalk and Elena Demidova. HapPenIng: Happen, Predict, Infer — Event Series Completion in a Knowledge Graph. International Semantic Web Conference (ISWC), 2019. (Full paper) [GD19b]
- Simon Gottschalk, Viola Bernacchi, Richard Rogers, Elena Demidova. Towards Better Understanding Researcher Strategies in Cross-Lingual Event Analytics. International Conference on Theory and Practice of Digital Libraries (TPDL), 2018. (Full paper) [GD18a]
- Simon Gottschalk and Elena Demidova. EventKG: A Multilingual Event-centric Temporal Knowledge Graph. Extended Semantic Web Conference (ESWC), 2018. (Resource paper) [GD18a]
- Simon Gottschalk, Nicolas Tempelmeier, Günter Kniesel, Vasileios Iosifidis, Besnik Fetahu and Elena Demidova. Simple-ML: Towards a Framework for Semantic Data Analytics Workflows. International Conference on Semantic Systems (SEMANTiCS), 2019. (Short paper) [GTK+19]

Workshop papers

- Sara Abdollahi, Simon Gottschalk and Elena Demidova. EventKG+Click: A Dataset of Language-specific Event-centric User Interaction Traces.

Workshop on Cross-lingual Event-centric Open Analytics co-located with the Extended Semantic Web Conference (CLEOPATRA@ESWC), 2020 (Full workshop paper) [AGD20]

- Simon Gottschalk, Endri Kacupaj, Sara Abdollahi, Diego Alves, Gabriel Amaral, Elisavet Koutsiana, Tin Kuculo, Daniela Major, Caio Mello, Gullal S. Cheema, Abdul Sittar, Swati, Golsa Tahmasebzadeh and Gaurish Thakkar. OEKG: The Open Event Knowledge Graph. Workshop on Cross-lingual Event-centric Open Analytics co-located with The Web Conference (CLEOPATRA@TheWebConf), 2021 (Full workshop paper, Best Paper Award) [GKA+21]

Poster and demonstration papers

- Simon Gottschalk and Elena Demidova. EventKG+BT: Generation of Interactive Biography Timelines from a Knowledge Graph. Extended Semantic Web Conference (ESWC), 2020. (Demonstration paper) [GD20]
- Simon Gottschalk and Elena Demidova. EventKG+TL: Creating Cross-Lingual Timelines from an Event-Centric Knowledge Graph. Extended Semantic Web Conference (ESWC), 2018. (Demonstration paper) [GD18b]
- Simon Gottschalk, Elena Demidova, Viola Bernacchi, Richard Rogers. Ongoing Events in Wikipedia: a Cross-lingual Case Study. Conference on Web Science (WebSci), 2017. (Extended abstract) [GDBR17]
- Hady Elsahar, Elena Demidova, Simon Gottschalk, Christophe Gravier, Frederique Laforest. Unsupervised Open Relation Extraction. Extended Semantic Web Conference (ESWC), 2017. (Short paper) [EDG+17]
- Simon Gottschalk and Elena Demidova. Analysing Temporal Evolution of Interlingual Wikipedia Article Pairs. Conference on Research and Development in Information Retrieval (SIGIR), 2016. (Demonstration paper) [GD16]

Under submission

- Tab2KG: Transforming Tabular Data into Knowledge Graphs using Semantic Domain Profiles. Simon Gottschalk and Elena Demidova

Contents

List of Figures

List of Tables

1 Introduction

1.1 Motivation

The world keeps changing every minute, as a result of a multitude of events. So does the data that describes the world. Access to this data and the understanding of the represented knowledge is a prerequisite for a vast amount of tasks in everyone's life. Examples are the use of weather forecasts and online encyclopedias or conversations with virtual assistants. None of these applications may work without an understanding of world knowledge: the collection of weather statistics alone does not make a weather forecast, but an understanding of the involved concepts does.

A common concept underlying knowledge-based applications is the use of *knowledge graphs*, where real-world objects and their relations are represented as a graph of nodes and edges [HBC+20]. In a knowledge graph, such nodes and edges underlie semantic annotations, which enables machines to understand and reason over the represented knowledge. For example, Washington, D.C. $\xrightarrow{\text{capital of}}$ United States does represent knowledge about the United States and can be extended by many more nodes and edges. Semantic annotations could now declare Washington, D.C., as a city (Washington, D.C. $\xrightarrow{\text{type}}$ City) and require each city to have coordinates or the number of inhabitants. Methods that interact with knowledge graphs benefit from the represented knowledge and its semantics and open up a whole range of knowledge-based applications.

The question of how to represent knowledge in a logical and formalised way goes back to the 3rd century (with the suggestion of the Tree of Porphyry [Sow12]) and has received growing attention in the last century, with the introduction of existential graphs [Pei09] and conceptual graphs [Sow76], amongst others. These early works have in recent years attributed to the creation of knowledge graphs such as DBpedia [ABK+07], Freebase [BEP+08], YAGO [SKW07] and Wikidata [VK14], which are nowadays well established and used in a broad range of applications. Examples of applications based on these knowledge graphs include DBpedia Spotlight for natural

language understanding [MJGSB11], IBM Watson for answering questions posed in natural language with the help of YAGO [FBCC+10], and the vision of creating a multilingual Wikipedia, i.e., an encyclopedia based on a language-independent abstraction of the facts in Wikidata [Vra20].

Knowledge graphs are a representation of the real world. Any changes in the real world are initiated by events [Mat37], where the most significant ones are perceived by large communities of people and reported by the media [DK92]. As analyses have shown, events have a social impact – be it cultural events [DJ10], sports events [HM+06], or natural disasters [Alb18]. Thus, events influence everyone's life. Such influence and the perception of the events can heavily vary across communities [Rog13]. A knowledge graph which is specifically designed to represent event knowledge is an essential step towards event-centric analytics and exploration of events and their impact in the world.

The representation of events in a knowledge graph adds new dimensions and poses several new challenges. First of all, time takes an important role, given the temporal nature of events. Second, events are very heterogeneous, both concerning their characteristics and concerning the information available for the same event in different sources (for instance, compare a natural eruptive event such as an earthquake and a long-lasting, human-made event such as the Brexit). Third, there is a need to set a limit of what to insert into a knowledge graph. The decision of what is an event of historic nature and hence supposed to be added to an event knowledge graph is a crucial question, as already discussed decades ago [Mat37]. While every action of each individual may be regarded as an event, only those which are viewed as specifically influential to some audience will make their way to a focused event knowledge graph. These challenges demonstrate the importance of creating an event knowledge graph which provides a semantic understanding of events.

The creation of a knowledge graph, i.e., the representation of knowledge in an integrated schema, is a task which heavily depends on the present data and the given scenario: (i) Integration of knowledge from existing sources: knowledge is spread across the web in a large variety of formats, including textual data, semi-structured sources and already existing knowledge graphs. The selection and extraction of data from such sources and their integration into a knowledge graph is a common but challenging task [WT10]. (ii) Creation of a knowledge graph from user-given data: For data analysis, users often require deep analytics of just a specific fraction of data, which they provide as an input to data analytics workflows. The transformation of such data into a knowledge graph, and thus the provision of semantics, is an important factor for making data analytics workflows more robust and efficient [GTK+19]. (iii) Enrichment of an existing knowledge graph: In most cases, knowledge graphs do not represent closed knowledge and are incomplete by nature, which calls for methods to enrich the represented knowledge. One way of doing so is the inference of knowledge which is already implicitly given in the knowledge graph itself [Pau17].

Access to a knowledge graph – wherever it comes from – does not imply immediate

and intuitive access to the knowledge it represents: most potential users are not familiar with the concepts of knowledge graphs, not to mention query languages and the specific schema. Therefore, methods for intuitive access to knowledge is a crucial task to make a knowledge graph usable for a broader audience [MKG+18]. Such methods vary depending on the user needs: Users may want to take a close look at single objects in a knowledge graph or want to start exploration from a much broader viewpoint, for instance, the dependencies between several objects without a formulated search intent to start with [WR09]. Approaches to tackling such requirements not only need to provide an intuitive visualisation as an entry point to the knowledge but also need to make a selection of what knowledge is worthy of showing to the user in the respective scenario [ADM+15].

Figure 1.1 gives an example how an event makes its way from the real world to end-user applications: At first, something happens which affects a large community of people, as the inauguration of Barack Obama as US president. Such information is modelled as part of a knowledge graph. When users later ask a search engine for related information, they may indirectly profit from such knowledge graph. In this example, Google returns a well-structured list of US presidents and their terms in office as a reply to the particular user query "list of US presidents". These examples demonstrate that knowledge graphs are present in everyone's life – even if the end-users do not know what they are [Her16].

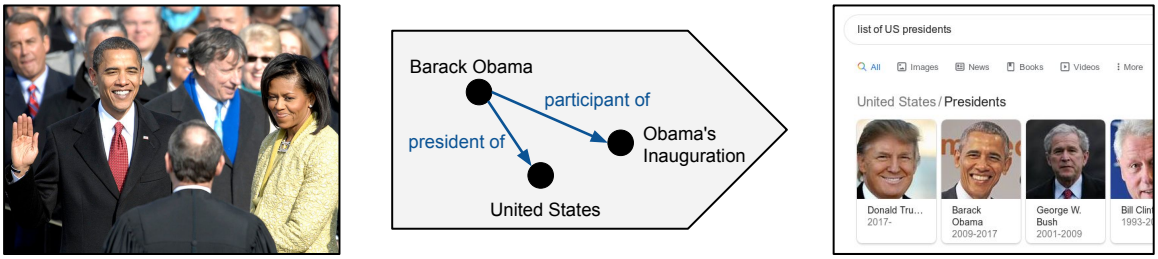


Figure 1.1. An example of how a real-world event (Barack Obama's inauguration) is first represented in a knowledge graph and how such knowledge can later be employed by an end-user application (Google's search engine).[1]

1.2 Research Questions

The path from collecting and representing knowledge to the provision of intuitive access to such knowledge involves several challenges which are addressed in this thesis.

[1]The photo on the left is taken from Wikimedia Commons (https://commons.wikimedia.org/wiki/File:US_President_Barack_Obama_taking_his_Oath_of_Office_-_2009Jan20.jpg) and in the public domain in the United States. The screenshot on the right was taken from https://www.google.com/search?q=list+of+us+presidents in July 2020.

Based on these challenges, we derive four research questions which are introduced in this section.

Knowledge Graph Creation

Obviously, the creation of a knowledge graph is the foundation of any upcoming knowledge-based task. Therefore, we put a special focus on this issue. As already explained before, the creation of an event knowledge graph is particularly challenging. While event knowledge is present in several sources, it remains a challenge of how to extract that knowledge from the sources and how to integrate it. This challenge directly leads to our first research question:

- **RQ1.1** How to create an event knowledge graph that integrates knowledge from several sources?

Existing knowledge graphs already contain event knowledge: For example, YAGO 3.1 has 392,844 events and DBpedia currently has 77,583 named events such as the Second World War and the FIFA World Cup 2014. This makes evident that knowledge graphs vary in their coverage of events and the question of which nodes in the graph are actually typed as events. In addition, they miss out important but unlabelled events such as "WikiLeaks co-founder Julian Assange is arrested after seven years in Ecuador's embassy in London" which are typically found in news articles or semi-structured sources such as Wikipedia's Current Events Portal[2]. To bring such heterogeneous sources together, **RQ1.1** implies the need for creating a common knowledge graph schema which integrates those different perspectives. **RQ1.1** also calls for the *fusion* of conflicting information: Input sources may disagree with specific facts, as for example the happening time of a long-lasting event.

In the current scenario, we assume control over the incoming data that is transformed into a new event-centric knowledge graph. But what if we do not know what given data is about, but we still want to profit from semantic interpretation?

We envision a scenario where a user wants to analyse a dataset given in tabular format, for example, a table of football matches, their attendances and betting odds. Without any semantic interpretation, data analysis tools can only view such a table as a set of text strings and numbers. Consequently, they miss out the opportunity to view such values as actual semantic concepts. In this thesis, we will investigate how an automated semantic interpretation of tabular data is possible, under the assumption of the availability of underlying background knowledge:

- **RQ1.2** How to create a knowledge graph from tabular data with the help of background knowledge?

[2] https://en.wikipedia.org/wiki/Portal:Current_events

We see background knowledge as observations from previously seen datasets in the specific domain. Consider the football example from above: a column of numerical values between $5,000$ and $80,000$ may be learnt to represent the attendance of football matches, while betting odds are in a range between 1 and 10. Now if a user asks for the semantic interpretation of a table with previously unseen values, the respective concepts could be recognised following the background knowledge of value distributions.

After recognition of semantic concepts in the data, a new knowledge graph can be created, which reflects what is represented in the input table. This newly created knowledge graph will guide the subsequent steps of data analysis.

Enrichment of Knowledge Graphs

Typically, knowledge graphs follow the *open-world assumption*, which basically states that the non-existence of facts in the knowledge graph does not imply they do not hold in the real world. While this assumption makes knowledge graph creation flexible towards missing information, it also implies that knowledge graphs are incomplete by nature. Such characteristic calls for knowledge graph enrichment methods, which add new nodes or edges to a knowledge graph. In this thesis, we pursue the question of how to conduct knowledge graph enrichment in a setting without the availability of additional data sources:

- **RQ2** How to enrich a knowledge graph without relying on additional, external knowledge?

This research question implies that we seek for knowledge which is already implicitly contained in the knowledge graph, but not yet explicitly represented as part of it. As an example, consider a knowledge graph which contains the fact that Obama was the president of the United States, but misses the fact that he was a resident of the White House. However, this fact could be inferred from facts about other US presidents.

Application of Knowledge Graphs

All the previous research questions contribute towards the final goal of this thesis, which is to make knowledge graphs accessible to non-expert end-users. This is a challenging task as we can not assume that such user has any idea of what a (knowledge) graph is and how its information can be obtained. Even if a user was proficient in such concepts, the user might still be overwhelmed by the amount of information potentially identifiable in a knowledge graph. These challenges lead to the final research question tackled in this thesis:

- **RQ3** How to apply an event knowledge graph for providing event knowledge to an end-user?

Events are a particularly interesting example for knowledge graph applications: There are many aspects surrounding an event, and thus there is much to explore. Second, an event knowledge graph is temporal by nature, which opens up many possibilities for applications that have a focus on the temporal order of things, with a prominent example being timelines. However, the space of a timeline is limited. Therefore, **RQ3** is not just about creating an intuitive interface to a knowledge graph but also about the collection and filtering of data which actually fulfils the user needs.

1.3 Contributions

Figure 1.2 summarises the contributions reported in this thesis, divided into three steps: (i) knowledge graph creation, (ii) knowledge graph enrichment, and (iii) knowledge graph application. In combination, this pipeline follows a logical order – starting from input sources and ending in interactive demonstrators which are based on the created and enriched knowledge graphs. In detail, we present the following contributions in the remainder of this thesis:

1.3.1 Knowledge Graph Creation

Without surprise, the very first step on our way to knowledge graph-based applications deals with the creation of said knowledge graphs. Based on the research questions **RQ1.1** and **RQ1.2**, we create knowledge graphs in two different settings:

EventKG

In Chapter 3, we present *EventKG*, our temporal and event-centric knowledge graph. *EventKG* incorporates information extracted from several large-scale knowledge graphs such as Wikidata, DBpedia and YAGO, as well as less structured sources such as the Wikipedia Current Events Portal and Wikipedia event lists. To enable the integration of such amount of heterogeneous sources, we develop a schema, an extraction pipeline and methods for fusing conflicting input data.

In its current version 3.0, *EventKG* contains events in 15 languages and provides information for over 1.3 million events and over 4.5 million temporal relations. More than a half of the events (56.25%) originate from the existing knowledge graphs; the others are extracted from semi-structured sources.

Tab2KG

In Chapter 4, we introduce *Tab2KG*, our approach towards **RQ1.2**, i.e., the transformation of tables into knowledge graphs. With *Tab2KG*, we consider as background knowledge a *domain profile*, i.e., statistics with respect to a target schema. Given such

background knowledge, *Tab2KG* automatically infers the semantics of tabular data and transforms this data into a knowledge graph. We propose a one-shot learning approach that relies on these profiles to create a mapping between a tabular dataset that contains previously unseen instances and a target schema. In contrast to the existing approaches, *Tab2KG* relies on the profiles only and does not require direct access to any data instances in the background knowledge. Our experimental evaluation on several real-world datasets demonstrates that *Tab2KG* outperforms semantic labelling baselines by nine percentage points on average.

1.3.2 Knowledge Graph Enrichment

Our method *HapPenIng* described in Chapter 5 conducts knowledge graph enrichment based on event series.

HapPenIng

RQ2 asks for the enrichment of a knowledge graph without the use of external resources. We approach this task in a particular setting that opens up a new perspective into knowledge graph enrichment: Typically, enrichment tasks infer new relations between objects already represented in the knowledge graph. Instead of that, we propose a novel approach that infers new objects, i.e., our approach adds new nodes to the graph. This is challenging, as such objects are not known yet, and we need to infer their label and more. Again, event knowledge graphs serve as a great example to make such enrichment possible: They typically include *event series*, i.e., sequences of related events, as the annual Wimbledon Championship editions or the US presidential elections that happen every four years. The detection of patterns within event series allows us to identify missing editions of event series, and to infer not only their label but also their time and location.

HapPenIng performs knowledge graph enrichment for event series in two steps: First, it applies machine learning models for identifying missing sub-event relations between two events (e.g., the Wimbledon Championships final of 2018 is a sub-event of the Wimbledon Championships 2018) based on a set of textual and spatio-temporal features. Then, it leverages structural features of event series for inferring missing editions of an event series. As explained, *HapPenIng* does not require any external knowledge. Our experimental evaluation demonstrates that *HapPenIng* outperforms baselines by 44 and 52 percentage points in terms of precision for the sub-event prediction and the inference tasks, respectively.

1.3.3 Knowledge Graph Application

We present two types of event knowledge graph applications: biography timelines and event timelines.

Biography Timelines: *EventKG+BT*

Research on notable accomplishments and relevant events in the life of people of public interest usually requires close reading of long encyclopedic or biographical sources, which is a tedious and time-consuming task [GBRD18]. In Chapter 6, we demonstrate an application of *EventKG* to biography timeline generation, where we adopt a distant supervision method to identify relations most relevant for a biography of a person of public interest. Our results of a user study and an automatic evaluation demonstrate the effectiveness of the proposed method, and we show that our method significantly outperforms the TimeMachine system [ADM+15] for biography generation.

EventKG+BT is our interactive demonstrator of the generated biography timelines. With *EventKG+BT*, users get access to concise and interactive spatio-temporal representations of biographies and can start the exploration of any person of public interest that is included in *EventKG*.

Event Timelines: *EventKG+TL*

EventKG+TL is our interactive demonstrator of event timelines. It generates cross-lingual event timelines based on *EventKG* and facilitates an overview of the language-specific event relevance and popularity along with the cross-lingual differences. Thus, *EventKG+TL* is a tool for exploring events and their perception in different language communities. Analogous to *EventKG+BT*, *EventKG+TL* hides the event knowledge graph from the user and abstracts its knowledge to make it accessible for a broad audience.

1.4 Thesis Outline

The remainder of this thesis is organised as follows: In Chapter 2, we provide the foundations of concepts and methods relevant in the following chapters. This includes an introduction into knowledge graphs and the Resource Description Framework (RDF) (Section 2.1), a definition of events and an overview of event knowledge graphs (Section 2.2), a brief survey of knowledge graph creation and enrichment techniques (Section 2.3 - 2.4) and an overview of applications based on knowledge graphs, with a specific focus on timelines (2.5). This chapter also introduces a running example, which is used throughout this thesis.

The following four chapters (Chapter 3 - 6) provide details of our research, all framed with an introductory section, a problem statement, an evaluation and a discussion, plus a section of chapter-specific background knowledge. In Chapter 3, we introduce ***EventKG***, our event-centric knowledge graph. Amongst others, this chapter includes the definition of a schema and an extraction pipeline (Section 3.5), as well as dataset characteristics and an evaluation of the extraction process (Section 3.6). Chapter 4 introduces ***Tab2KG***, our approach for tabular data interpretation.

Within this chapter, we describe the *Tab2KG* approach in detail and explain the role of semantic profiles (4.5), following an evaluation section (4.6). Chapter 5 deals with ***HapPenIng*** – an approach for event knowledge graph enrichment based on event series. This approach is divided into two sub-tasks: sub-event prediction and event inference, both described in Section 5.4, also followed by an evaluation (Section 5.5). In Chapter 6, we demonstrate example applications of *EventKG* at the example of event and biography timelines. First, we describe and evaluate our approach to biography timeline generation (Section 6.5 - 6.6). Then, we give two examples of running systems for event-centric knowledge exploration: ***EventKG+BT*** (Section 6.7) and ***EventKG+TL*** (Section 6.8).

Finally, Chapter 7 gives a summary of this thesis and a discussion of the findings. The chapter closes with an outlook to future works.

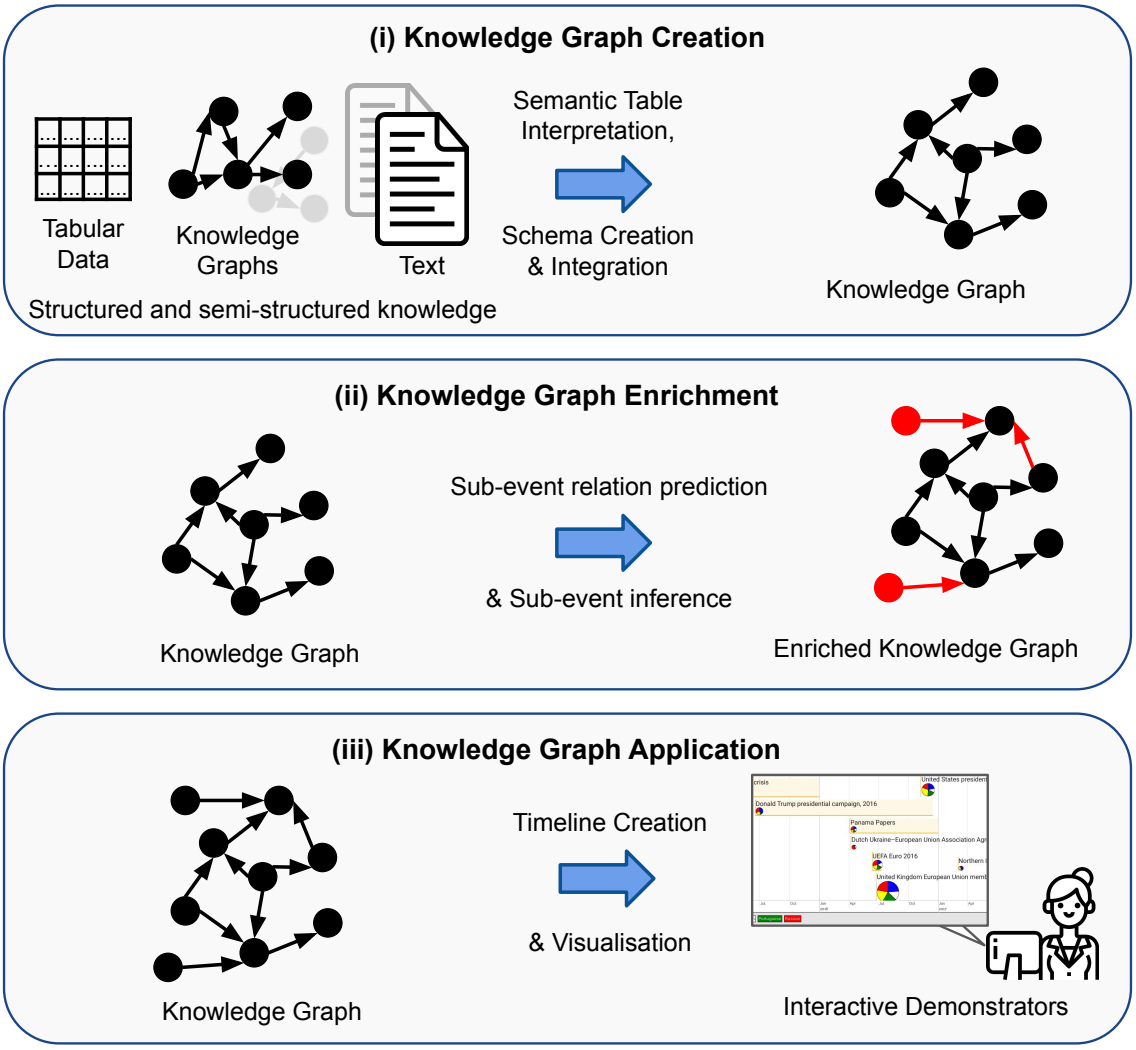


Figure 1.2. Overview of the contributions of this thesis. (i) Knowledge graphs are created from tabular data, other knowledge graphs and semi-structured data. (ii) At the example of an event knowledge graph, we demonstrate a method for knowledge graph enrichment, which is solely based on information inherent in the event knowledge graph. (iii) Finally, timelines are extracted from the knowledge graph and visualised with interactive demonstrators.

2 Background

In this chapter, we give an overview of relevant concepts in this thesis: knowledge graphs and events, as well as the creation, enrichment and application of knowledge graphs. We start with an introduction to knowledge graphs and how they are used for representing knowledge about real-world objects. Then, we define what events are and how they can be represented as part of a knowledge graph. In the following, we have a detailed look at how to create and enrich knowledge graphs from event data and tabular data. Finally, we describe what applications can be built on top of event knowledge graphs, with a particular focus on timelines.

2.1 Knowledge Graphs

Knowledge graphs[1,2] are a means to represent knowledge about the real world. When speaking of the real world, we refer to *entities*, which could be persons like Barack Obama, locations like Washington, D.C.[3], events such as the US Presidential Election in 2012, concepts such as Chemistry, and others. Such entities hold specific characteristics: for example, Barack Obama was born in 1961, and Washington has 672,228 inhabitants. Entities are related to each other via *properties*: Obama was the winner of the US Presidential Election in 2012 and was a resident of Washington. On top of that, knowledge graphs add *semantics* to the represented knowledge. Semantics define the meaning of high-level terms in a knowledge graph [HBC+20] and thus allow for the interpretation of data and the inference of new facts: For example, if we know that Obama was the president of the United States, we could infer that he was a politician.

Formally, a knowledge graph is defined as follows (based on [HBC+20]):

[1]The term "Knowledge Graph" was first used to refer to the Google Knowledge Graph [Sin12], but has since been adopted as a common name for similar datasets [MEC+18].

[2]A knowledge graph can also referred to as a graph-based *knowledge base*.

[3]We will abbreviate Washington, D.C., as Washington in the rest of this chapter.

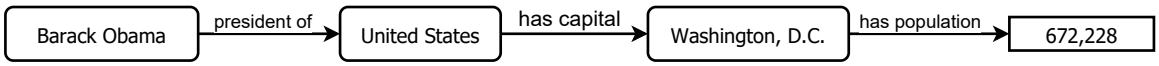


Figure 2.1. A first example of a knowledge graph about Barack Obama.

Definition 2.1. *A **knowledge graph** is a directed graph $G = (N, R)$, whose nodes N represent entities and literal values, and whose edges R represent relations between these entities.*

Following this definition, a knowledge graph is a directed, edge-labelled graph that represents entities in the real world and their relations, plus literal values (*literals* like texts and numbers). There are no limitations given to the covered entities, the edge labels or the domain covered in the knowledge graph. As a consequence, knowledge graphs can grow flexibly, and they can be specifically about a *domain* (such as medicine) or represent cross-domain knowledge. However, to uniformly interact with different knowledge graphs and to provide machines with an understanding of the semantics behind the entities and relations, a set of modelling rules and other techniques are required.

In the remainder of this thesis, we will introduce different instantiations of this knowledge graph definition, including a temporal knowledge graph, a data graph, a schema graph, and a domain knowledge graph.

2.1.1 Running Example: Barack Obama's Life Represented in a Knowledge Graph

Throughout this thesis, we will repeatedly come back to a concrete example that deals with the life of Barack Obama, the former president of the United States. Starting from the basics of knowledge representation using a knowledge graph, we will end up creating a timeline of significant events in his life. For a start, assume we want to model the fact that Obama was the president of the United States. In addition, we want to model context information about the United States, concretely about its capital Washington, and its population. Figure 2.1 shows a simple knowledge graph representing this knowledge: (i) There are three nodes, which represent the real-world entities Barack Obama, the United States and Washington. Also, there is a literal value ("672,228") which denotes the population of Washington. (ii) There are three edges, all directed and labelled that provide semantic connections between two entities. Each of them comes with a property label ("president of", "has capital" and "has population"). While the first two relations connect two entities, the latter one connects an entity to a literal value.

2.1.2 The Resource Description Framework (RDF)

Until now, we have formally defined knowledge graphs. This does, however, not yet explain how information represented as a knowledge graph can be represented in a machine-understandable way. This idea goes back to the vision of Time Berners-Lee in 1999 to create a *Semantic Web* which makes it possible for computers to "become capable of analysing all the data on the Web – the content, links, and transactions between people and computers" [BF00], based on "a language that expresses both data and rules for reasoning about the data" [BLHL01]. Such goals are enabled through the *Resource Description Framework* (RDF), which is designed as "a standard model for data interchange on the Web" [W3C06].

In RDF, knowledge graph nodes are defined as *resources* that are represented by *Uniform Resource Identifiers* (URIs) like `http://example.org/resource#Barack_Obama`. Relations between two resources are represented as *triples* consisting of a *subject*, a *predicate* and an *object*. Alternatively, relations can connect a subject resource to a literal value. Therefore, the subject of a triple is always a resource, but the object can be either a resource or a literal value.

There are different textual syntaxes for expressing RDF, of which we use the Terse RDF Triple Language (Turtle, TTL)[4] and N-Quads[5,6] in the remainder of this thesis.

Knowledge graphs represented in RDF can be queried through the *SPARQL Protocol and RDF Query Language* (SPARQL) [Con08].

Running Example

Listing 2.1 shows a set of RDF triples representing the example knowledge graph in Figure 2.1, using the TTL notation. Now, each node is uniquely identified by a URI, which is composed of a *namespace* and the *local name*. For example, `ex:BarackObama` refers to the URI `http://example.org/resource#Barack_Obama`, using the prefix `ex`, the corresponding namespace `http://example.org/resource#`, and the local name *Barack_Obama*.

Each remaining line in Listing 2.1 follows the triple structure. For example, the first triple has `exo:BarackObama` as its subject, `exo:presidentOf` as the predicate, and `exo:UnitedStates` as the object. The third triple comes with a literal value as the object.

[4] https://www.w3.org/TR/turtle/
[5] https://www.w3.org/TR/n-quads/
[6] N-Quads allow the declaration of multiple named graphs within one document (see Section 2.1.4).

Listing 2.1: A first example of the RDF notation

```
@prefix ex: <http://example.org/resource#> .
@prefix exo: <http://example.org/ontology#> .

ex:BarackObama      exo:presidentOf      ex:UnitedStates .
ex:UnitedStates     exo:hasCapital       ex:WashingtonDC .
ex:WashingtonDC     exo:hasPopulation    672228 .
```

2.1.3 Schemas and Ontologies

To add semantics to a knowledge graph, a *schema* or *vocabulary* defines and describes a set of terms, which facilitates reasoning over the knowledge graph [HBC+20]. Such terms include: (i) *Classes*: Typically, entities can be grouped into classes. For example, both the United States and Washington are locations, i.e., they can be modelled as *instances* of the class `Location`. Such an assignment of a resource to a class is done using the `rdf:type` property. Within the schema, a class hierarchy can be defined: For example, Washington can be modelled as `City`, which is a *subclass* of `Location`. A reasoner may infer now that Washington is a `Location`. (ii) *Properties*: Properties usually connect instances of specific classes. A schema can define the classes of the subject (*domain*) and the object (*range*) a property connects. Furthermore, a schema can also define property hierarchies. For example, the property `president of` can be modelled as *sub-property* of `head of state`. Its domain could be `Politician`, and its range could be `Country`. A common vocabulary for describing classes and properties is RDF Schema (RDFS) [BGM14] which offers properties such as `rdfs:subClassOf`, `rdfs:subPropertyOf`, `rdfs:domain` and `rdfs:range`.

Domain-specific knowledge graphs are used to represent knowledge of a particular *domain* (i.e., a specific subject area or area of knowledge [Hef04])[7], e.g., medicine, geography or events. To adequately represent such knowledge and to enable knowledge inference, there is a need to define an *ontology*, which includes computer-usable definitions of the classes in the domain and the relationships among them [Hef04]. To this end, an ontology language such as the OWL Web Ontology Language (OWL) [MVH04] is used. OWL provides a much richer vocabulary for describing classes and properties than RDFS, including cardinality of relations and negations. For example, the property `has capital` can be modelled as `owl:FunctionalProperty`. Under this condition, each country is only allowed to have exactly one capital.

[7] Note that the domains of properties and domains in the sense of knowledge areas are named the same, but they are two different concepts.

2.1.4 Additional Techniques for Knowledge Graph Representation in RDF

Figure 2.2 shows an extended version of the knowledge graph shown before in Figure 2.1, which covers several additional techniques for representing a knowledge graph using RDF. Still, the knowledge graph represents the real-world entities Barack Obama, United States and Washington, using URIs[8], labels and classes.

There are more techniques illustrated in Figure 2.2, that play an important role in the remainder of this thesis:

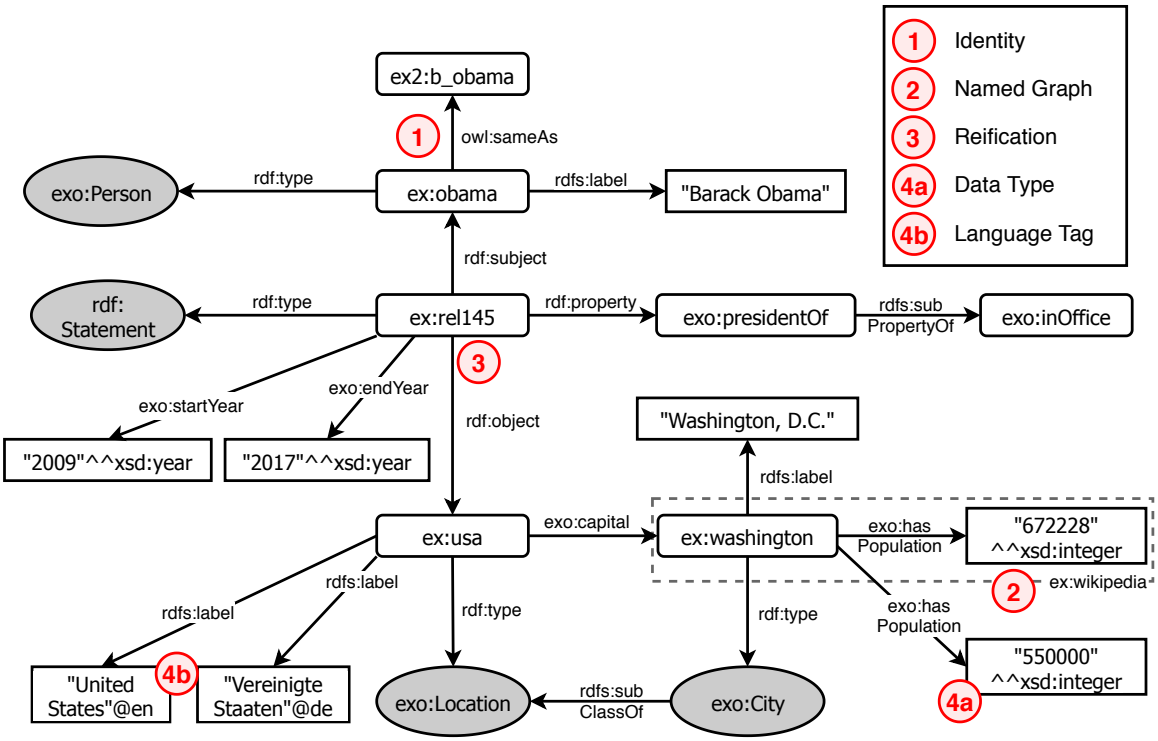


Figure 2.2. An extended version of the knowledge graph about Barack Obama. Some advanced techniques are highlighted with circled numbers.

1. **Identity**: Nodes in different knowledge graphs can refer to the same real-world entity. Such identify can be represented using the `owl:sameAs` property. Interlinking nodes across different knowledge graphs is one of the core principles of *Linked Data* [BL06]. In Figure 2.2, `ex:obama` and `ex2:b_obama` both represent Barack Obama and are linked accordingly.

2. **Named Graphs**: Statements in RDF can be identified using a *named graph*, which is also identified by an URI. In Figure 2.2, there are two different values

[8] This knowledge graph uses the exemplary prefixes `ex` and `ex2` for entities, `exo` for ontology terms, as well as `rdf`, `rdfs` and `xsd`, which are described later.

given for the population of Washington. One of them is assigned a source (`ex:wikipedia`), using a named graph.

3. **Reification**: RDF requires the representation of a knowledge graph in triples. However, not all relations in a knowledge graph are binary. For example, the provision of temporal context information for a given relation demands additional modelling techniques. One way of doing so is *reification*, where a new node is introduced that represents an edge. In addition to the edge's subject, property and object (via `rdf:subject`, `rdf:property` and `rdf:object`), more context can be added. In our example, the start and end year of Obama's presidency are attached to the node `ex:rel145`.

   Further solutions to model non-binary relations include named graphs to group together statements and RDF* where statements themselves can be treated as nodes [Har17].

4. **Quoted Literals**: In RDF, a literal value can be encapsulated in quotation marks and enriched with a data type or a language tag:

   (a) **Data Types**: The data type of a literal value (e.g., text, integer or date) can be expressed in RDF. In our example, the population is declared as integer using the `xsd` prefix, which is described in the next section. Properties with a data type as range are called *data type properties*, in contrast to *object properties*.

   (b) **Language Tags**: The language of textual literal values can be assigned using a language tag (e.g., `@en` or `@de`). In Figure 2.2, we use language tags to differ between the German and the English label of the United States.

2.1.5 Selected Knowledge Graphs and Vocabularies

As of May 2020, the Linked Open Data Cloud[9] contained 1,255 datasets "that have been published in the Linked Data format", in a variety of domains. In a similar fashion, the Linked Open Vocabularies[10] linked to 714 vocabularies. In this section, we introduce selected knowledge graphs and vocabularies that are utilised in the remainder of this thesis.

Table 2.1 gives an overview of three popular, free and cross-domain knowledge graphs, which also play an essential role in this thesis. These knowledge graphs depend on other sources, e.g., human input, WordNet [Mil98] (a lexical database) and Wikipedia (the well-known free and user-generated encyclopedia[11]). It is important

[9] https://lod-cloud.net/
[10] https://lov.linkeddata.es/dataset/lov/
[11] www.wikipedia.org

Table 2.1. Selected cross-domain knowledges graphs.

Name	Release Year	Self Description
DBpedia [ABK+07]	2007	Crowd-sourced community effort to extract structured content from the information created in various Wikimedia projects.[12]
YAGO [SKW07]	2008	Huge semantic knowledge base, derived from Wikipedia WordNet and GeoNames.[13]
Wikidata [VK14]	2012	Free and open knowledge base that can be read and edited by both humans and machines.[14]

Table 2.2. Selected vocabularies.

Name	Prefix	Self Description
RDF	`rdf:`	RDF is a standard model for data interchange on the Web.[15]
RDF Schema[16]	`rdfs:`	Data-modelling vocabulary for RDF data.[17]
XML Schema	`xsd:`	A language for expressing constraints about XML documents[18], whose data type definitions are used in RDF.[19]
DBpedia ontology	`dbo:`	Shallow, cross-domain ontology, which has been manually created based on the most commonly used infoboxes within Wikipedia.[20]
Schema.org	`so:`	Collaborative, community activity with a mission to create, maintain, and promote schemas for structured data on the Internet, on web pages, in email messages, and beyond.[21]
Data Catalog Vocabulary	`dcat:`	RDF vocabulary designed to facilitate interoperability between data catalogues published on the Web.[22]

to note that Wikipedia, although in itself only semi-structured [VKV+06], takes an important role for many knowledge graphs. For instance, DBpedia does only contain resources which are also covered in Wikipedia, and they even share the same local names. Similarly, all entities covered by Wikipedia are also represented as a node in Wikidata.

Table 2.2 lists five particularly relevant vocabularies: RDF, RDF Schema (RDFS), the DBpedia ontology, Schema.org, and DCAT, the Data Catalog Vocabulary [C+14]. More vocabularies are used in the remainder of this thesis, including the Simple Event Model (`sem:`), the Semantic Sensor Network Ontology (`sosa:`), and Wikidata's property namespaces (`wdt:`).

[12] https://wiki.dbpedia.org/about
[13] https://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/yago
[14] https://www.wikidata.org/wiki/Wikidata:Main_Page
[15] https://www.w3.org/TR/rdf-schema
[16] https://www.w3.org/TR/rdf-schema/
[17] https://www.w3.org/TR/rdf-schema
[18] https://www.w3.org/standards/xml/schema
[19] https://www.w3.org/TR/swbp-xsch-datatypes/
[20] https://wiki.dbpedia.org/services-resources/ontology

2.2 Events and Event Knowledge Graphs

> *"Any past event is historical,*
> *but only the most memorable ones are historic."*
>
> — William Safire, 1992

Events, with examples including military conflicts (e.g., World War II), sports tournaments (e.g., Wimbledon Championships 2018) and political elections (e.g., US presidential election in 2012), take on a special role in the context of knowledge graphs. As explained until now, knowledge graphs capture knowledge that can be represented using statements and existing ontologies. The heterogeneity, variety in cultural perception, and dynamics of events pose several challenges when it comes to defining what an event is and when modelling an event as part of a knowledge graph.

2.2.1 Events

There is a wide variety in the definitions of what makes an event: From a physical point of view, an event can be simply seen as a "change of state" [Mat37], which does not give any restrictions to the event characteristics. On the contrary, Dayan and Katz discuss that events are "interruptions of routine" and it is their media coverage that makes them events [DK92]. Contemporary dictionaries follow a much more generic definition of an event as "something that happens"[23]. In computer science, such definitions are extended with specific event characteristics: Following Allan, an event is "something that happens at a particular time and place" [APL98]. Another important aspect is the societal significance of an event ("a thing that happens or takes place, especially one of importance" [Dic89]), which is also expressed by the quote of William Safire at the beginning of this section. In this thesis, events are defined as real-world happenings of societal importance:

Definition 2.2. *An **event** is something of societal importance that happened in the real world.*

Following this definition, we both ensure societal importance of events and do not put any restrictions on the structure or characteristics of events. Instead, we define a set of optional event characteristics, including but not limited to:

- Label: *Named events* have a unique label, e.g., "Second inauguration of Barack Obama" and "Wimbledon Championships 2018".

[21] https://schema.org/
[22] https://www.w3.org/TR/vocab-dcat-2/
[23] Marriam Webster Dictionary: https://www.merriam-webster.com/dictionary/event

- Location: Most[24] events happen at one or more specific locations, e.g., the second inauguration of Barack Obama happened in Washington.
- Time: Events happen at a specific time or time interval. In some cases, such time is clearly identifiable (e.g., the day of an inauguration). In the case of longer-lasting or ongoing events (e.g., the Brexit), it can be difficult to identify the correct time interval.
- Participants: Events involve a set of participants, which can be persons, organisations, and other entities.

Figure 2.3 gives an example of five events which all show different characteristics (in clockwise order): (i) The second inauguration of Barack Obama follows the definition by Allan et al. [APL98] of an event as something that happens at a particular time and place. Beyond that, more information is given, such as the event participants. (ii) The Wimbledon Championships is an annually held tennis tournament, i.e., an *event series*, which has a set of *editions*, e.g., the Wimbledon Championships 2018. (iii) In contrast to the former two event examples, the arrest of Julian Assange in 2019 does not come with a unique label and is instead represented by a sentence, potentially extracted from a news article. (iv) The labels and descriptions of events of local or regional importance (for example the opening of the Hannover trade fair) are often not available in English. (v) The marriage between Barack and Michelle Obama is modelled as a relation with a validity time span, i.e., as a *temporal relation*. Such relation takes an important role in Barack Obama's life and indirectly refers to their wedding in 1992.

2.2.2 Event Knowledge Graphs

As an *event knowledge graph*, we consider an event-centric temporal knowledge graph, i.e., a knowledge graph which represents events and their characteristics, as well as temporal relations between entities and events. As shown in Figure 2.3, the creation of such an event knowledge graph requires the integration of several different types of events, which requires a joint event ontology and integration process.

Selected Event Vocabularies

Several data models and the corresponding vocabularies (e.g., [RvEV+16, VHMS+11, Guh11, STH09, Sch12])[25] provide means to model events. For example, the ECKG model proposed by Rospocher et al. [RvEV+16] enables fine-grained textual annotations to model events extracted from news collections. The Conflict and Mediation Event Observations framework (CAMEO) [Sch12] is a framework to encode events

[24] Exceptions include virtual events such as virtual conferences.
[25] For a broader overview of event ontologies, refer to [STH09] and [VHMS+11].

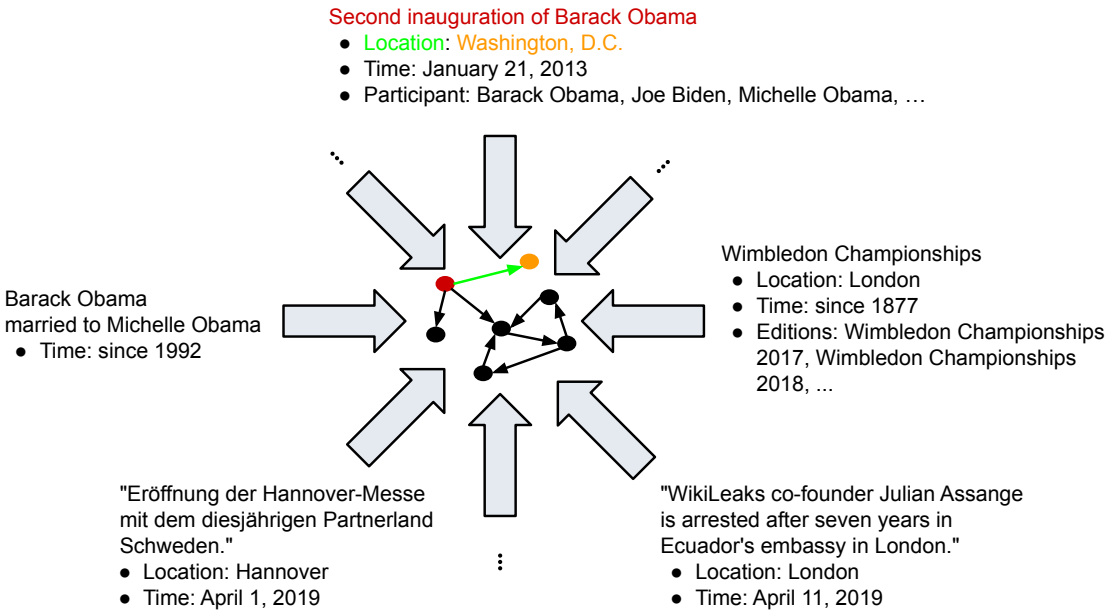


Figure 2.3. Examples of different events and their representation in an event knowledge graph.

extracted from the news, in particular in the political domain, using a specific event taxonomy. The Simple Event Model (SEM) [VHMS+11], the schema.org vocabulary [Guh11] which focuses on events such as concerts, lectures and festivals, the Linking Open Descriptions of Events (LODE) ontology [STH09], and the CIDOC Conceptual Reference Model (CRM) [Doe03] provide means to describe events and interlink them with participants, times and places. Figure 2.4 shows the most relevant classes and properties of the Simple Event Model: It models events (`sem:Event`), event participants (`sem:Actor`), locations (`sem:Place`) and time spans (`sem:hasBeginTimeStamp` and `sem:hasEndTimeStamp`).

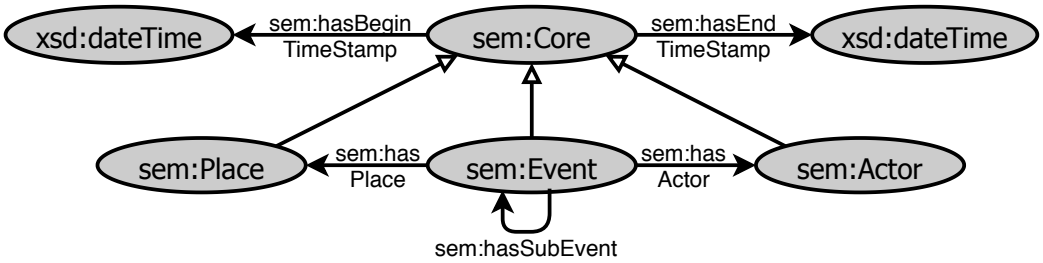


Figure 2.4. Excerpt of the Simple Event Model (SEM). $\twoheadrightarrow$ marks a `rdfs:subClassOf` relation, $\rightarrow$ denotes the domain and range of a property.

Selected Event Knowledge Graphs

We now present a selection of event knowledge graphs. Given the heterogeneity of event representations, these datasets vary a lot, and we divide them into three groups:

Annotated news article streams. Following Dayan [DK92], events are particularly represented in media. Consequently, there exist several datasets which extract events directly from news streams. The Global Data on Events, Location and Tone (GDELT) [LS13] and the Integrated Conflict Early Warning System (ICEWS) [BLO+15] both are large-scale datasets (GDELT has 326 million event mentions between January 2015 and February 2016[26]) of actions between two entities, found in news articles and encoded using the CAMEO framework mentioned before. Thus, these datasets cover mainly events from the political domain and provide several annotations, but no event labels or descriptions. A similar approach is taken by the Event Registry [LFBG14], where events are represented as clusters of news articles, categorised and annotated, e.g., with links to entity representations in Wikidata. Common to these datasets is that they annotate news articles as they come in, without consideration of the significance of the covered events.

Named events contained in knowledge graphs. Although not event-specific in particular, cross-domain knowledge graphs such as those presented in Section 2.1.5 contain named events such as the Second World War and the Wimbledon Championships 2018. For example, YAGO has 392,844 resources typed as `schema:Event` and DBpedia has 77,583 instances of `dbo:Event`. A common problem with the event representation in such sources is that they are tied to their specific ontologies and demand structural information of their covered events, limiting their event coverage [RvEV+16].

Semi-structured sources. Between processing whole news articles and providing single named event resources, there is the case of having short event descriptions. One example of such event representations is the Wikipedia Current Events Portal[27] where users collect "brief summaries of the topic at hand, preferably no more than 30 - 40 words", where topics are of "international interest" and "big at the moment"[28]. Efforts have been made to convert the Wikipedia Current Events into a machine-readable format [HWP12, TA14]. However, in many cases, events from semi-structured sources can only hardly be represented using existing event ontologies and thus cannot be properly combined with existing knowledge graphs.

2.3 Knowledge Graph Creation

The creation of a knowledge graph means extraction and representation of data, more concrete to discover and canonicalise entities and their semantic types and

[26] https://blog.gdeltproject.org/the-datasets-of-gdelt-as-of-february-2016/
[27] https://en.wikipedia.org/wiki/Portal:Current_events
[28] https://en.wikipedia.org/wiki/Wikipedia:How_the_Current_events_page_works

organizing them into clean taxonomies [WDRS20]. This process heavily depends on its sources, which could provide structured, semi-structured or unstructured data. Subject to the sources, a variety of different tasks can be involved in knowledge graph creation, including machine learning, natural-language processing and data integration [WT10, SWW+15]. In this thesis, we focus on two cases of knowledge graph creation: the creation of event knowledge graphs from known semi-structured and structured sources as well as the creation of knowledge graphs from previously unseen tabular data.

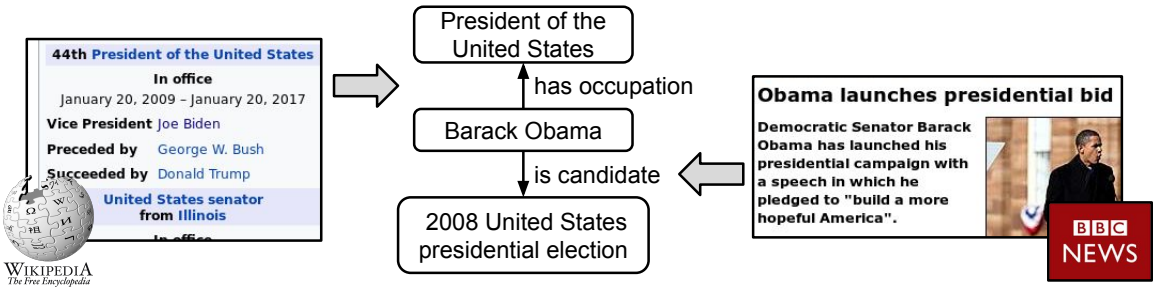


Figure 2.5. Example of knowledge graph creation. On the left side, a fact is extracted from the Wikipedia infobox about Barack Obama. On the right side, a fact is extracted from unstructured text in a news article[29].

Figure 2.5 gives an example of knowledge graph creation from a structured source and an unstructured source.

2.3.1 Knowledge Graph Creation from Structured Sources

Structured web sources used for knowledge graph extraction include Wikipedia categories [WT10], the Wikipedia Current Events Portal [HWP12, TA14] and social media platforms [FIND18]. Another prominent example are the Wikipedia infoboxes, which are present alongside Wikipedia articles, follow pre-defined templates and contain facts about the entity represented in the respective Wikipedia article. Figure 2.5 shows an example of an infobox with facts about Barack Obama. Wikipedia infoboxes and categories have always been the main focus of YAGO, where information about entities extracted from these sources are combined with the classes of WordNet [TWS20]. In general, knowledge graph creation from structured sources benefits from the pre-known structures but obviously lacks all the knowledge on the web which is not represented in a structured way.

[29] http://news.bbc.co.uk/2/hi/americas/6349081.stm

2.3.2 Knowledge Graph Creation from Unstructured Sources

Knowledge graph extraction from plain text such as news articles has been addressed in a considerable number of works, many focusing on events [ABBC+17, RvEV+16, LS13, BLO+15, WKGS19, SWW+15]. These approaches apply open information extraction methods (i.e., the identification and classification of previously unseen relations between entities [EDG+17]) and develop them further to address specific challenges of event extraction from news. State-of-the-art approaches that automatically extract events from the news potentially obtain noisy and unreliable results (e.g., the state-of-the-art extraction approach in [RvEV+16] reports an accuracy of only 0.55). Furthermore, such systems provide billions of events at a very high granularity level, as typically represented in news articles. Compared to the established knowledge graphs such as YAGO, DBpedia or Wikidata, such events indicate significant differences in the representation accuracy and event granularity.

2.3.3 Knowledge Graph Creation from Tabular Data

When referring to structured sources, we have assumed that their structures are known before starting the extraction process. For example, we may know that a specific Wikipedia infobox contains all facts related to a particular person, who is the subject of all extracted facts. Here, we assume no knowledge about the structure: all that is given is a table plus background knowledge about the respective domain, e.g., football or weather data. This is a typical scenario when running data analytics [GTK+19, TKSA16a].

Knowledge graphs as a means to add machine-readable semantics to any kind of data can serve the purpose of making data analytics workflows more efficient, robust and reusable [GTK+19]. One factor for doing so is the use of a machine learning vocabulary that enables semantic configuration of machine learning [EMN+15]. In this thesis, we focus on the first part of data analytics workflows, which is the preparation of data. Consider Figure 2.6, where a tabular dataset of political events is transformed into a knowledge graph.

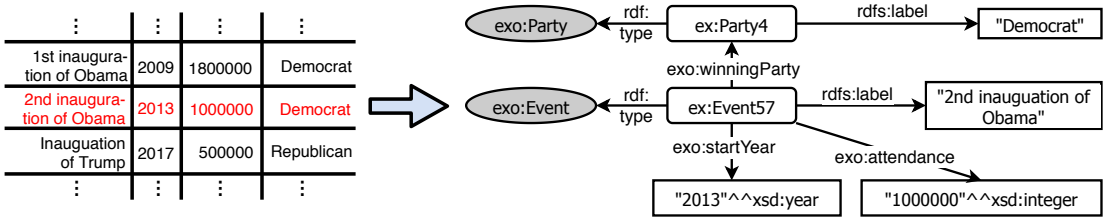


Figure 2.6. Example of a tabular dataset and the knowledge graph it is transformed to. The marked row in the center is transformed into the knowledge graph shown on the right.

Ontology-based data access (OBDA) is a term first coined in 2008 by Poggi et al. [PLC+08] that is predominantly used in the field of relational databases. In OBDA systems, an ontology serves as a domain-specific abstraction layer that abstract away from the way data is stored. Based on said ontology, users can query the relational data using SPARQL queries [XCK+18]. An OBDA system translates SPARQL queries into the SQL language for accessing data stored in a database (query translation), or by translating the data into RDF format (data translation) [CPCF20]. An important tool for doing so are mapping languages such as the RDF Mapping Language (RML), which describe mapping rules from heterogeneous data to RDF [DVSC+14]. When wrapping the data sources and the query processing with distributed methods, OBDA can even handle large and heterogeneous data sources [MGS+19].

OBDA requires the availability of a mapping between data sets and an ontology. In other words, there is a need to understand the semantics behind the different data units, such as the columns in a data table. The process of gaining such understanding is called *semantic table interpretation*, *semantic labelling* or *semantification*. Concretely, semantic table interpretation is typically defined as the task that takes a data table and a knowledge graph as input, and returns a semantically annotated table as output [CRSDP19].

Recently, semantic table interpretation has often been split into three subtasks, as defined in the SemTab challenge [JRHE+20]:

- Column-Type Annotation (CTA) describes the mapping of all values in a table column to a class in an ontology. In the example in Figure 2.6, this would be the annotation of the first column with `exo:Event`.
- Cell-Entity Annotation (CEA) deals with the linking of single table cell values to a resource in a knowledge graph. In the example, this could be the annotation of the cell "Democrat" with the DBpedia entity dbpedia.org/resource/Democratic_Party_(United_States), for example.
- Columns-Property Annotation (CPA) assigns a property to the relationship between two columns. In the example, this is the annotation of the relationship between the first and fourth column with the property `exo:winningParty`.

Semantic table interpretation methods involve several approaches, including entity lookup and voting strategies [NKIT19, CJRHS19], feature-based entity matching at the instance-level [CDPRS20, NKIT19] and user interaction [KSA+12, CCDPP19]. Based on the semantic annotations returned by the semantic table interpretation (i.e., the mapping of columns to classes and properties in the ontology), the input data table can be transformed into a knowledge graph, as shown in Figure 2.6.

2.3.4 Knowledge Fusion

With the extraction of knowledge from different sources comes the challenge of data integration, which has been an active research area in the field of databases for several decades [Len02]. In the case of knowledge graph creation, we deal with the specific case of *knowledge fusion*, i.e., the identification of "true subject-predicate-object triples extracted [...] from multiple information sources" [DGH+15]. Knowledge fusion approaches include majority voting, quality-based methods that assess the trustworthiness of sources [DGH+15], and embedding-based probabilistic approaches [DGH+14].

2.3.5 Dataset Profiling

After the creation of a knowledge graph, it is important to make it accessible, for example, as part of the Linked Open Data Cloud. This requires following standards such as the aforementioned core principles of Linked Data [BL06] and the FAIR principles [WDA+16] for making data findable, accessible, interoperable, and reusable. Most of these principles can be attributed to the provision of metadata or a *dataset profile*.

A dataset profile is a formal representation of a set of dataset features, where a dataset feature is a characteristic describing a certain attribute of the dataset [BEBB+18]. Dataset features can be arranged in a taxonomy of general, qualitative, provenance, links, licensing, statistical and dynamics categories [BEBB+18], for example including the following ones:

- Statistical features, such as the size and the average number of triples in the dataset, as well as the property co-occurrences.
- Provenance features that allow to track down the origins of the data.
- Licensing features, i.e., the type of license under which the dataset can be used.

Dedicated vocabularies can be used to describe a dataset profile in RDF, such as the Data Catalog Vocabulary (`dcat`) introduced in Table 2.2 (with properties such as `dcat:downloadURL`) and the VoID vocabulary [ACHZ11] (with properties such as `void:triples` to denote the number of triples in a dataset).

Data profiling is important to dataset retrieval where users select corpora based on specific criteria. In [KA13], the authors propose a user query approach to retrieve relevant RDF datasets by applying semantic filters to a set of available datasets. [NP19] demonstrates how to enable spatio-temporal search over Open Data catalogues through the creation of a spatio-temporal knowledge graph. In our proposed framework, dataset retrieval is enabled both through the semantic dataset profiles as well as through a domain model.

2.4 Knowledge Graph Completeness & Enrichment

Upon creation, knowledge graphs are usually not complete, which calls for knowledge graph enrichment.

2.4.1 Knowledge Graph Completeness under the Open-World Assumption

Knowledge graphs usually follow the *open-world assumption* which "assumes only the information given in the [knowledge graph] and hence requires all facts, both positive and negative, to be explicitly represented", in contrast to many databases that follow the *closed-world assumption*, where a "negative fact is implicitly present provided its positive counterpart is not explicitly present" [Rei81]. As an example, consider the initial example knowledge graph shown in Figure 2.1: Under the closed-world Assumption, there have not been other US presidents than Barack Obama. Under the open-world assumption, the non-existence of the statement `George Washington, president of, United States` does not imply the falseness of this statement. Consequently, there can be more US presidents, although not (yet) included in the knowledge graph.

Completeness is an essential dataset quality dimension [BEBB+18]. However, due to the open-world assumption, knowledge graphs are notoriously incomplete [RSN16, TPS+17]. There has been research on several exemplary aspects of knowledge graph completeness, for example, on the incompleteness of Wikidata [BRN18, ARP17] and the relation between obligatory attributes and missing attribute values [LS18]. These works emphasise the need for knowledge graph completion, in particular regarding event-centric information.

2.4.2 Knowledge Graph Enrichment

The task of adding missing information to a knowledge graph has been given several names, including knowledge graph enrichment, knowledge graph completion and knowledge graph refinement. We will use the term *knowledge graph enrichment* in the remainder of this thesis. Paulheim [Pau17] identifies three different knowledge graph enrichment approaches which we will explain in the following. For illustration, we come back to the first example given in this chapter (Figure 2.1), which can be enriched with new nodes and edges, as shown in Figure 2.7.

1. **Type Assertions Completion**. Type assertions completion is the task of predicting a type or a class of an entity [Pau17]. A common approach to this task is to probabilistically exploit type information that is inherent in the statement properties [PB13]. For example, assume that there are more persons represented in the knowledge graph shown in Figure 2.7, of which a large number

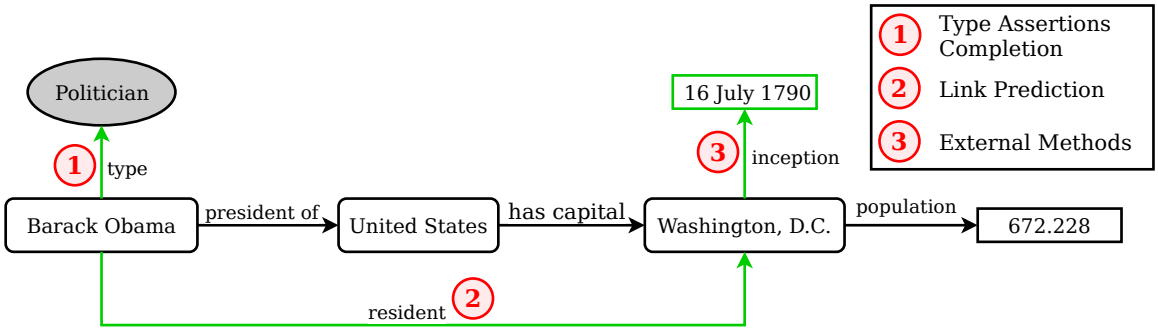


Figure 2.7. Examples of three different knowledge graph completion approaches. New nodes and edges are marked in green.

with a `president of` relation is typed as `Politician`, while most others are not. From that observation, one could infer that also Barack Obama is a politician.

2. **Link Prediction**. In the case of link prediction, a ranked list of subjects, predicates or objects is generated, given an incomplete triple. Approaches to solving this task are typically based on embeddings, i.e., lower-dimensional representations of the knowledge graphs, which are still preserving their structural information and characteristics [CZC18]. Examples include the TransE [BUGD+13], STransE [NSQJ16] and other graph embedding models [SW17, WMWG17]. In Figure 2.7, there is a new relation (`resident`) added between Barack Obama and Washington. This could be the result of link prediction if it infers that US presidents always reside in Washington.

3. **External Methods**. Information extraction approaches can be used to detect new edges [RGP17] and nodes [KVW14] from external textual data. An example is shown in Figure 2.7, where the inception date of Washington is added to the knowledge graph, extracted from an external source.

None of the knowledge graph completion and refinement tasks have yet considered the inference of new nodes given only the knowledge graph itself [WMWG17, Pau17]. In Chapter 5, we generate new events not initially present in the knowledge graph without the use of external sources.

2.5 Application of Knowledge Graphs

Access to the knowledge represented in a knowledge graph requires different levels of expertise: A user needs to understand the concept of knowledge graphs, must be proficient in the query language and needs to know the underlying ontology. Also, a user might be overwhelmed with the potentially large amount of results for a specific information need [ADM+15]. These limitations call for applications which lessen the

burden of accessing and exploring knowledge represented in a knowledge graph and provide easily understandable visualisations [GBRD18].

Methods to intuitively access semantic information included in knowledge graphs include question answering (i.e., the translation of natural-language queries into a query language) [HWM+17, HZLL19, CGD20], spatio-temporal search applications [ZCZ+17, HWM+17, HZLL19, NP19] and interactive query construction interfaces [DZN12, DZN13]. Such methods focus on the construction of queries, but less on the exploratory nature of user indents which go beyond pure lookup tasks [WR09]. Given a clear setting (i.e., a user who explores historical paintings and their artists), there are plenty of ways to interact with knowledge graphs. For example, consider the four applications shown in Figure 2.8[30], which are all based on Wikidata and serve different needs: The openArtBrowser [Hum20] allows exploration of artists and their works, Scholia [NMW17] presents statistics about the research of scientists, ViziData[31] provides a spatio-temporal visualisation of selected event categories, and Wikidata Graphs[32] displays query results on a timeline.

2.5.1 Timelines

In this thesis, we focus on *timelines* as a specific application of knowledge graphs, where a timeline is a list of chronologically sorted *timeline entries*. Timelines help to identify the relevant information that is often "buried in an avalanche of data" [ADM+15] and can be used in several scenarios (for example, to identify linguistic points of view [SLW+17]). Examples of timelines are (i) *event timelines* which provide relevant sub-events or related events, given an entity or event of user interest [VCK15, Els16], and (ii) *biography timelines* which show relevant events in the lifetime of a person of interest [ADM+15, TEPW11]. Figure 2.9 gives an example of a biography timeline about Barack Obama, which is created from the TimeMachine demonstrator [ADM+15], on top of the Freebase knowledge graph [BEP+08].

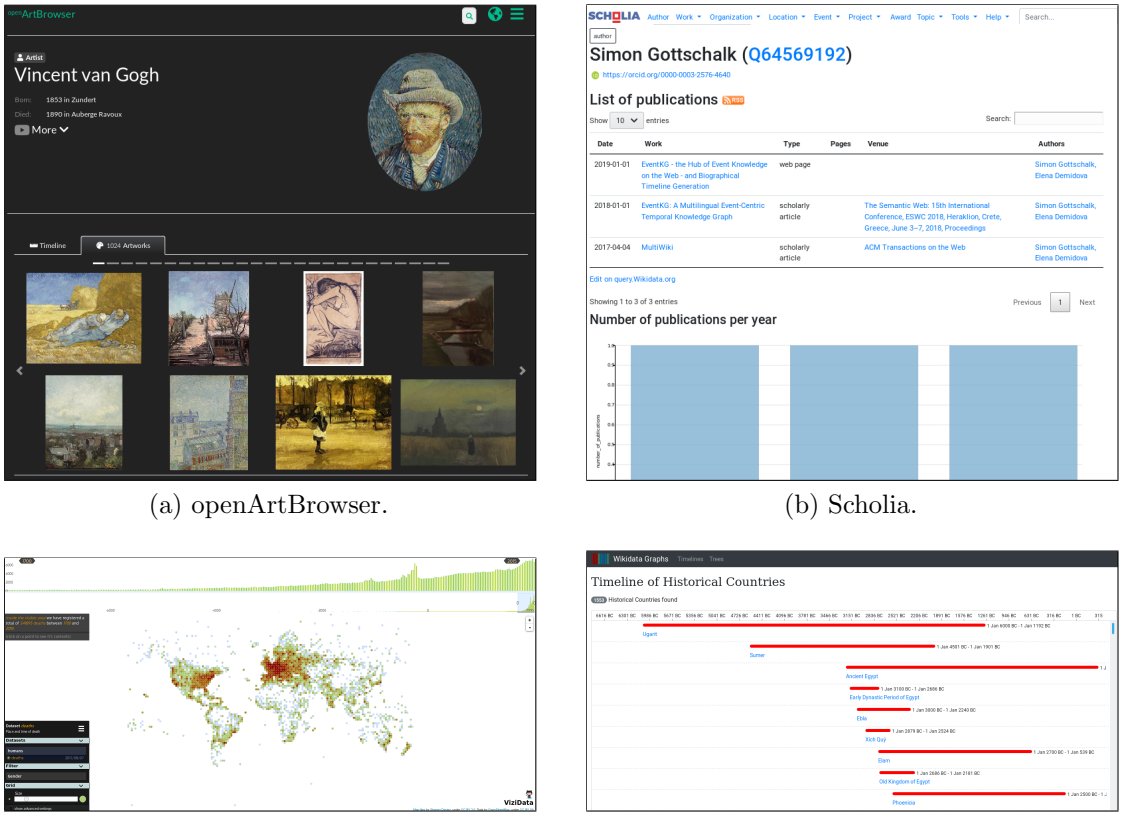
Common to both event and biography timelines are the following two tasks: (i) Extraction of candidate timeline entries which are connected to the query entity, and (ii) identification of relevant timeline entries. While the first task involves access to the knowledge graph, the second task reduces the available information such that a user can still grasp the presented knowledge without being overwhelmed by the given wealth of knowledge.

[30] The four screenshots were taken from https://openartbrowser.org/artist/Q5582, https://scholia.toolforge.org/author/Q64569192, http://sylum.lima-city.de/viziData and https://wikidata-graphs.herokuapp.com/timeline/historical-countries in July 2020.

[31] https://sylum.lima-city.de/viziData/

[32] https://wikidata-graphs.herokuapp.com

[33] This screenshot was taken from https://cs.stanford.edu/~althoff/timemachine/demo.html in July 2020.



(a) openArtBrowser. (b) Scholia.

(c) ViziData. (d) Wikidata Graphs.

Figure 2.8. Example applications based on the Wikidata knowledge graph.

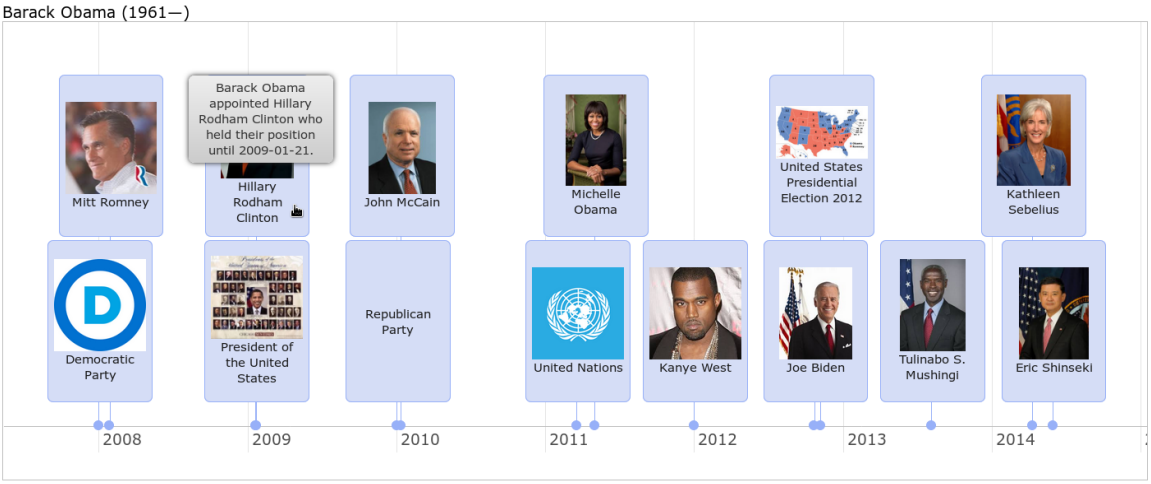


Figure 2.9. An example of a biography timeline about Barack Obama given by the TimeMachine [ADM+15]. The timeline displays important entities in the life of Barack Obama in chronological order. By hovering over a specific entity, a user gains more information about the specific relation.[33]

3 Creation of an Event Knowledge Graph

Without surprise, the existence of a knowledge graph is a prerequisite for any knowledge graph-based process or system. Such a knowledge graph reflects the knowledge which is of interest in the respective setting. As we explained in Chapter 1 and 2, event knowledge is of major interest in a large number of application scenarios. As there was no knowledge graph specifically targeting to cover event knowledge, we create an event knowledge graph based on several sources such as other knowledge graphs that do contain event knowledge but do not treat events and temporal information as first-class citizens in their schemas. The goal of creating an event knowledge graph goes along with our research question **RQ1.1**, which asks about the creation of an event knowledge graph from several sources. This chapter will give an answer to this question, which is *EventKG*: a multilingual event-centric temporal knowledge graph.

3.1 Introduction

The amount of event-centric information regarding contemporary and historical events of global importance, such as the US elections, the 2018 Winter Olympics and the Syrian Civil War, constantly grows on the Web, in the news sources and within social media. Efficiently accessing and analysing large-scale event-centric and temporal information is crucial for a variety of real-world applications in the fields of Semantic Web, Natural Language Processing and Digital Humanities. In Semantic Web and Natural Language Processing, these applications include timeline generation [ADM+15, GD18b] and Question Answering [HWM+17, HZLL19]. In Digital Humanities, multilingual event repositories can facilitate cross-cultural studies analysing language-specific and community-specific views on historical and contemporary events (examples of such studies can be seen in [GDBR17, Rog13]). Furthermore, event-centric knowledge graphs can facilitate the reconstruction of histories as well as networks of people and organisations over time [RvEV+16, ABBC+17]. One of the pivotal pre-requisites to facilitate effective analytics of events is the availability of knowledge repositories

providing reference information regarding events, involved entities and their temporal relations.

As described in Section 2.2, an event is typically described as something that happens at a specific time and location [APL98]. In this thesis, we consider as events real-world happenings of societal importance (Definition 2.2), with examples including military conflicts, sports tournaments and political elections. In particular, we consider events, entities they involve and temporal relations – i.e., real-world relations between events and entities valid over a time period. Currently, event representations and temporal relations are spread across heterogeneous sources. First, large-scale knowledge graphs (e.g., Wikidata, DBpedia and YAGO, as introduced in Section 2.1.5) typically focus on entity-centric knowledge. Event-centric information included in these sources is often not clearly identified as such, can be incomplete and is mostly restricted to named events and encyclopaedic knowledge.

For example, as discussed later in Section 3.6.1, out of 322,669 events included in *EventKG* V1.1, only 18.7% are classified using the `dbo:Event` class in the English DBpedia as of 12/2017. Furthermore, event descriptions in the existing knowledge graphs often lack the key properties, i.e., times and locations. For example, according to our analysis, only 33% of events in Wikidata provided temporal and 11.7% spatial information at that time.

Second, a variety of manually curated semi-structured sources (e.g., the Wikipedia Current Events Portal (WCEP) [TA14] and multilingual Wikipedia event lists) contain information on contemporary events. However, the lack of structured representations of events and temporal relations in these sources hinders their direct use in real-world applications, e.g., through semantic technologies. Overall, a comprehensive, integrated view on contemporary and historical events and their temporal relations is still missing. *EventKG* will help to overcome these limitations.

An additional source of event-centric information on the Web are knowledge graphs containing events obtained from unstructured news sources using Information Extraction methods (such as [RvEV+16, YRH+18, PKN18, LS13, BLO+15]). These knowledge graphs are potentially highly noisy [RvEV+16]. Due to significant differences in quality and event granularity, the integration of events from these sources with the information in the established knowledge repositories such as DBpedia or Wikidata within a common knowledge graph does not appear meaningful. These event sources, as well as the corresponding Information Extraction methods for unstructured news articles, are out of the scope of this work.

In this chapter, we formalise the concept of a temporal knowledge graph that interconnects real-world entities and events using temporal relations valid over a time period. Furthermore, we present an instantiation of a temporal knowledge graph – *EventKG*. *EventKG* takes an important step to facilitate a global view on events and temporal relations currently spread across entity-centric knowledge graphs and manually curated semi-structured sources. *EventKG* integrates this knowledge in an efficient light-weight fashion, enriches it with additional features such as indications

of relation strengths and event popularity, adds provenance information and makes all this information available through a canonical RDF representation. Through the light-weight integration and fusion of event-centric and temporal information from different sources, *EventKG* enables to increase coverage and completeness of this information. For example, *EventKG* increases the coverage of locations and dates for the Wikidata events it contains by 14.43% and 17.82%, correspondingly (see Table 3.9 in Section 3.6.1 for more detail). Furthermore, relation strengths and event popularity provided by *EventKG* are the characteristics that gain the key relevance given the rapidly increasing amount of event-centric and temporal data on the Web and the resulting information overload.

EventKG, including the dataset, a SPARQL endpoint, the code and evaluation data, are available online[1].

Contributions. Overall, our contributions in this chapter are as follows:

1 We formally define the concept of a temporal knowledge graph TKG that incorporates entities, events and temporal relations.

2 We present an instantiation of the temporal knowledge graph TKG: *EventKG* – a multilingual RDF knowledge graph that incorporates more than 1.3 million events and more than 4.5 million temporal relations in version V3.0.

3 We provide insights into the extraction and fusion methods adopted to generate the *EventKG* knowledge graph and their quality.

Outline. The remainder of this chapter is organised as follows: First, in Section 3.2, we motivate the need for a temporal knowledge graph. Then, in Section 3.3, we provide connections to important concepts that were introduced in Chapter 2 and introduce more specific background where necessary. In Section 3.4, we formally define the concepts of a temporal knowledge graph. Then, in Section 3.5, we describe *EventKG*, including its RDF data model and the extraction pipeline. In Section 3.6, we provide statistics and evaluation results of the data contained in *EventKG*. This includes an overview of the updates that happened between *EventKG*'s versions V1.1 and V3.0. Finally, we discuss our findings and provide a conclusion in Section 3.7.

3.2 Motivation

Our society faces an unprecedented number of events that impact multiple communities across language and community borders. In this context, the efficient access to event-centric multilingual information originating from different sources, as facilitated by *EventKG*, is of utmost importance for several scientific communities, including Semantic Web, NLP and Digital Humanities and a variety of applications, as timeline

[1] http://eventkg.l3s.uni-hannover.de/

Table 3.1. All events connected to Barack Obama in *EventKG* that started between November 4 and November 16, 2011.

Start Date	Sources	Description
Nov 4	YAGO, Wikidata, $\text{DBpedia}_{\text{EN}}$, $\text{DBpedia}_{\text{FR}}$, $\text{DBpedia}_{\text{RU}}$	2011 G20 Cannes summit
Nov 11	YAGO, Wikidata, $\text{DBpedia}_{\text{EN}}$	2011 White House shooting
Nov 16	$\text{Wikipedia}_{\text{EN}}$	The President of the United States Barack Obama visits Australia to commemorate the 60th anniversary of the ANZUS alliance.

Table 3.2. Most linked events in the English (EN) and the Russian (RU) Wikipedia.

Rank	Event (EN)	#Links (EN)	Event (RU)	#Links (RU)
1	World War II	189,716	World War II	25,295
2	World War I	99,079	World War I	22,038
3	American Civil War	37,672	October Revolution	7,533
4	FA Cup	20,640	Russian Civil War	7,093

generation, question answering, as well as cross-cultural and cross-lingual event-centric analytics.

Timeline generation is an active research area [ADM+15, GD18b, GD20], where the focus is to generate a timeline (i.e., a chronologically ordered selection) of events and temporal relations for entities from a knowledge graph. In Chapter 6, we focus on the application of *EventKG* to the automated generation of timelines representing people biographies. In this task, information regarding event popularity and relation strength available in *EventKG* in combination with a benchmark extracted from external biographical sources can enable the selection of the most relevant timeline entries. In the same chapter, we will also show event timelines that particularly make use of *EventKG*'s language-specific relations.

At the example of timelines, we can see that *EventKG* contains complementary information originating from different reference sources, potentially resulting in more complete timelines and event representations. For example, Table 3.1 illustrates an excerpt from the timeline for the query *"What were the events related to Barack Obama between November 4 and November 16, 2011?"* generated using *EventKG*. The last event in the timeline in Table 3.1 about Obama visiting Australia extracted from an English Wikipedia event list ("2011 in Australia"[2]) is not contained in any of the reference knowledge graphs used to populate *EventKG* (Wikidata, DBpedia, and YAGO). The reference sources of the other two events include complementary information. For example, while the "2011 White House shooting" is assigned a start date in Wikidata, it is not connected to Barack Obama in that source.

[2] https://en.wikipedia.org/wiki/2011_in_Australia

Table 3.3. Top-4 persons mentioned jointly with the financial crisis (2007–2008) per language.

	EN	FR	DE	RU	PT
1	Barack Obama	Kevin Rudd	Barack Obama	Michael Moore	Barack Obama
2	George W. Bush	John Howard	Geir Haarde	Roman Abramovich	José Sócrates
3	Joseph Stiglitz	Don Cheadle	George W. Bush	Adam McKay	Pope Benedict XVI
4	Ben Bernanke	Ben Bernanke	Wolfgang Schäuble	Mikhail Prokhorov	Gordon Brown

An important application of *EventKG* are cross-cultural and cross-lingual analytics. Such analytics can provide insights into the differences in event perception and interpretation across communities. For example, event popularity and relation strength between events and entities varies across different cultural and linguistic contexts. These differences can be observed and analysed using the information provided by *EventKG*. For example, Table 3.2 presents the top-4 most popular events in the English vs the Russian Wikipedia language editions as measured by how often these events are referred, i.e., linked to in the respective Wikipedia language edition. Whereas both Wikipedia language editions mention events of global importance, here the two World Wars, most frequently, the other most popular events (e.g., "October Revolution" and "American Civil War") are language-specific. The relation strength between events and entities in specific language contexts can be inferred by counting their joint mentions in Wikipedia. For example, Table 3.3 lists the persons most related to the financial crisis in the years 2007 and 2008 in different Wikipedia language editions.

This information is directly provided by *EventKG*. As mentioned before, applications of *EventKG* are presented in Chapter 6, where we introduce two interactive systems: *EventKG+BT* for biography timelines and *EventKG+TL* for language-specific event timelines.

Another intended future application of *EventKG* is semantic event-centric question answering. With the provision of *EventKG*, it becomes possible to answer questions such as *"Which events related to Bill Clinton happened in Washington in 1980?"* and *"What are the most important events related to Syrian Civil War that took place in Aleppo?"* that are of interest for both cross-cultural and cross-lingual event-centric analytics (e.g., illustrated in [Rog13, GBRD18]) as well as question answering and semantic search applications (e.g., [HWM+17, ZCZ+17, HZLL19, DZN13]). *Event-QA* is a dataset with 1,000 semantic queries and the corresponding English, German and Portuguese verbalisations for answering event-centric questions over *EventKG* [CGD20]. For investigating user interaction traces in cross-lingual settings, *EventKG+Click* [AGD20] is a dataset that builds upon *EventKG* and language-specific information on user interactions with events, entities, and their relations derived from the Wikipedia clickstream. VisE-D builds an event type hierarchy on top of *EventKG* to support visual event classification [MBSH+21].

3.2.1 Running Example: Barack Obama

In Section 2.1.1, we have introduced our running example of representing Barack Obama's life in a knowledge graph. In this chapter, we will repeatedly provide examples of how *EventKG* represents events and facts related to Barack Obama. First, we will illustrate the heterogeneity of data about Barack Obama available in the reference knowledge graphs used to populate *EventKG* (Wikidata, DBpedia, YAGO and Wikipedia), and the extraction and integration of this data into a canonical RDF representation in *EventKG*. Later on, in Chapter 6, we will subsume these example and explicitly tackle the task of biography timeline generation at the example of Barack Obama's life.

3.3 Specific Background

In this section, we provide more specific background in the areas of event knowledge graphs.

Event Knowledge Graphs: To the best of our knowledge, currently, there are no dedicated knowledge graphs aggregating event-centric information and temporal relations for historical and contemporary events directly comparable to *EventKG*. The heterogeneity of data models and vocabularies for event-centric and temporal information (e.g., [STH09, RvEV+16, VHMS+11, Guh11, PKN18, YRH+18], as presented in Section 2.2.2), the large scale of the existing knowledge graphs, in which events play only an insignificant role, and the lack of clear identification of event-centric information, makes it particularly challenging to identify, extract, fuse and efficiently analyse event-centric and temporal information and make it accessible to real-world applications in an intuitive and unified way. Through the light-weight integration and fusion of event-centric and temporal information from different sources, *EventKG* enables to increase coverage and completeness of this information. Furthermore, existing sources lack structured information to judge event popularity and relation strength as provided by *EventKG* – the characteristic that gains the key relevance given the rapidly increasing amount of event-centric and temporal data on the Web and the resulting information overload.

In *EventKG*, we build upon SEM [VHMS+11] shown in Figure 2.4 and extend this model to represent a broader range of temporal relations and to provide additional information regarding events.

Extracting event-centric and temporal information: Most approaches for automatic knowledge graph construction and integration focus on entities and related facts rather than events. Examples include DBpedia [LIJ+15], Freebase [BEP+08], YAGO [MBS14] and YAGO+F [DON13]. In contrast, *EventKG* is focused on events and temporal relations. In [TA14], the authors extract event information from the Wikipedia Current Events Portal (WCEP). *EventKG* builds upon this work to include

WCEP events. For the extraction of temporal information, there are several approaches to annotate both textual data [KSSW16] and relations [RPN+14, TWM12] with temporal scopes inferred from external sources. In *EventKG*, we rely on the temporal information already contained in the reference sources, which gives highly precise values, as shown in Section 3.6.2. Increasing the coverage for temporal annotations in case of missing values by using external resources is a potential extension for future work.

The question of how to model temporal data is important as it comes to considering time expressions of different levels of granularity or with uncertainty. Examples to tackle such issues include the use of multiple potential start and end times as in the temporal slot filling task [Sur13] or adding uncertainty scores to temporal relations [CPSS17]. The representation of this information is facilitated through existing relational models [Che17], the Extended Date-Time Format (EDTF) [EDT] or with the Time Ontology in OWL [HP06]. The Simple Event Model adopted in this work supports a simple notion of temporal time spans, which is sufficient to represent temporal information provided by the reference sources of *EventKG* and is compatible with the time representation in these sources. Nevertheless, we see more advanced time models as a potential future extension, in particular in the context of a possible enrichment of *EventKG* with additional, and in particular, automatically inferred temporal information. For example, *EventKG* V3.0 adds granularity information to temporal literals.

Extraction of events and facts from the news: Recently, the problem of building knowledge graphs and datasets directly from plain text news articles [ABBC+17, RvEV+16, LS13, BLO+15], and extraction of named events from news [KVW14, YRH+18] have been addressed. These approaches apply Open Information Extraction methods and develop them further to address specific challenges in the event extraction in the news domain. State-of-the-art approaches that automatically extract events from news potentially obtain noisy and unreliable results (e.g., the state-of-the-art extraction approach in [RvEV+16] reports an accuracy of only 0.551).

Furthermore, such systems provide billions of events at a very high granularity level, as typically represented in news articles. Compared to the established knowledge repositories such as DBpedia or Wikidata, such events indicate significant differences in the representation accuracy and event granularity. In contrast, contemporary events included in *EventKG* originate from high-quality community curated sources such as WCEP and Wikipedia event lists and represent significant societal happenings at a different granularity and abstraction level, compared to news sources.

3.4 Problem Statement

A temporal knowledge graph G_T connects real-world entities and events using temporal relations, i.e., relations valid over a time period.

Definition 3.1. *A **temporal knowledge graph** $G_T = (E_t, R_t)$ is a directed multi-graph. The nodes in $E_t = E \cup \mathcal{V}$ are temporal entities, where E is a set of real-world entities and $\mathcal{V}$ is a set of real-world events. The directed edges in R_t represent temporal relations of the temporal entities in E_t.*

A temporal entity $e \in E$ represents a real-world entity such as a person, a location, an organisation or a concept. A temporal entity $e \in \mathcal{V}$ represents a real-world historical or contemporary event. Examples of events include cultural, sporting or political happenings. The temporal entities in G_T are characterised through their existence time (for real-world entities) or happening time (for events).

Definition 3.2. *A **temporal entity** $e \in E_t$ represents a real-world entity or event. e is annotated with a tuple $\langle\ e_{uri},\ e_{time}\ \rangle$, where e_{uri} is the unique entity identifier, and $e_{time} = [e_{start}, e_{end}]$ denotes the existence time of the entity (for $e \in E$) or the happening time of the event (for $e \in \mathcal{V}$).*

A temporal entity $e \in E_t$ can be assigned further properties, such as an entity type, a label and a textual description.

A temporal relation is a binary relation of the temporal entities valid over a certain period of time. More formally:

Definition 3.3. *A **temporal relation** $r \in R_t$ represents a binary relation between two temporal entities. r is annotated with a tuple $\langle r_{uri}, r_{time}, e_i, e_j \rangle$, where r_{uri} is a unique relation identifier, e_i and e_j are the temporal entities participating in the relation r and $r_{time} = [r_{start}, r_{end}]$ denotes the validity time interval of the temporal relation.*

The relation identifier r_{uri} reflects the semantics of the temporal relation and is typically specified as a vocabulary term.

3.5 *EventKG*: Approach

EventKG is a knowledge graph that instantiates the temporal knowledge graph defined in Definition 3.1, and at the same time facilitates the integration and fusion of a variety of heterogeneous event representations and temporal relations extracted from several reference sources.

A *reference source* is a semantic source such as a knowledge graph (e.g., Wikidata or YAGO) or a collection of articles (e.g., the French Wikipedia) used to populate *EventKG*.

In Section 3.5.1, we present the RDF data model of *EventKG* and its transformation into a temporal knowledge graph (Section 3.5.2). Following that, we present the *EventKG* generation pipeline in Section 3.5.3 and illustrate the pipeline steps with our running example of Barack Obama in Section 3.5.4.

3.5.1 *EventKG* RDF Data Model

The goals of the *EventKG* RDF data model are to facilitate a light-weight integration and fusion of heterogeneous event representations and temporal relations extracted from the reference sources, as well as to make this information available to real-world applications through an RDF representation. The *EventKG* data model is driven by the following objectives:

- Define the key properties of events through a canonical representation.
- Represent temporal relations between events and entities (including event-entity, entity-event and entity-entity relations).
- Include information quantifying and further describing these relations.
- Represent relations between events (e.g., in the context of event series).
- Support an efficient light-weight integration of event representations and temporal relations originating from heterogeneous sources.
- Provide provenance for the information included in *EventKG*.

EventKG schema and the Simple Event Model: In *EventKG*, we build upon the Simple Event Model (SEM) [VHMS+11] as a basis to model events in RDF. SEM is a flexible data model that provides a generic event-centric framework.

The main rationale of SEM is to provide a simple model that can represent events and their key properties. Events within *EventKG* come from heterogeneous sources where they can be described at a different level of detail. SEM provides the lowest common denominator for event-centric information, whereas it still includes the key properties of events and their relations. The properties of events in the *EventKG* data model are not mandatory, such that we can also include under-specified events in *EventKG*, e.g., in case the corresponding temporal or geospatial information is missing in the reference sources.

In addition to SEM, within the EvenKG schema, we adopt additional properties and classes to adequately represent the information extracted from the reference sources, to model temporal relations, and event relations as well as to provide provenance information. The schema of *EventKG* is presented in Figure 3.1 and the used RDF namespaces are listed in Table 3.4.

EventKG is an RDF-based dataset, so extensions to its data model are easily possible. In future work, such extensions can be performed to model confidence and uncertainty in the information extraction, integration and fusion, or to provide more fine-granular time information (e.g., using the EDTF (Extended Date-Time Format) [EDT]).

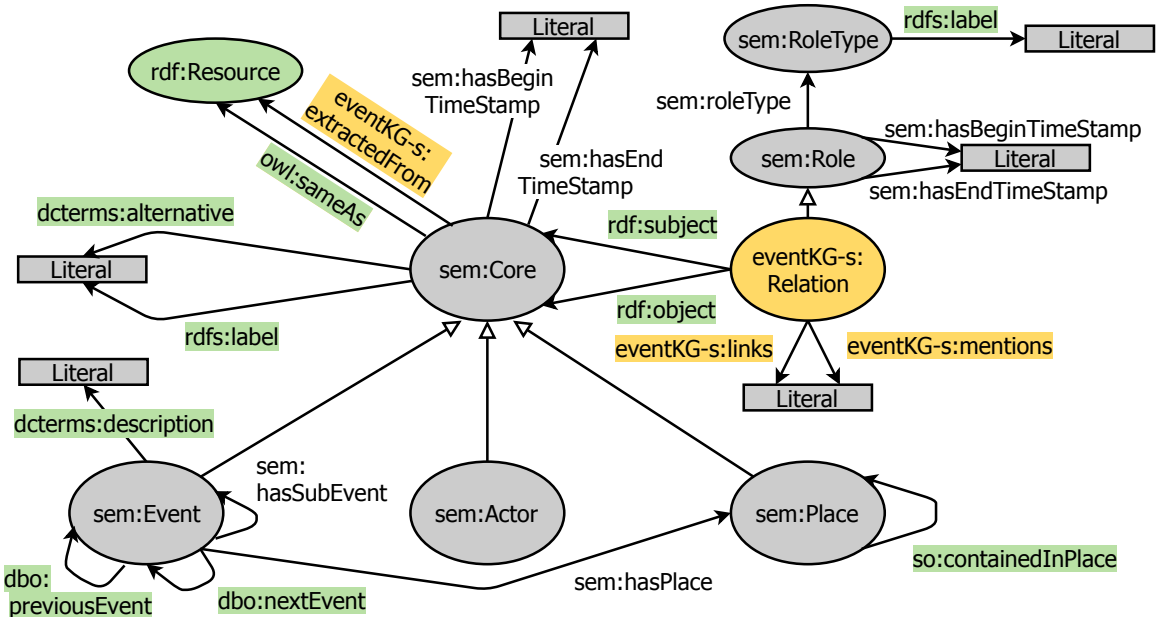


Figure 3.1. The *EventKG* schema based on SEM. Arrows with an open head denote `rdfs:subClassOf` properties. Regular arrows visualise the `rdfs:domain` and `rdfs:range` restrictions on properties. Terms from other reused vocabularies are coloured green. Classes and properties introduced in *EventKG* are coloured orange.

Events and entities: SEM provides a generic event representation including topical, spatial and temporal dimensions of an event, as well as links to its actors (i.e., entities participating in the event). Such resources are identified within the namespace `eventKG-r`. Thus, the key classes of SEM and the *EventKG* schema are `sem:Event` representing events, `sem:Place` representing locations and `sem:Actor` representing entities participating in the events (see Figure 2.4). Each of these classes is a subclass of `sem:Core`, which is used to represent all entities in the temporal knowledge graph[3]. Events are connected to their locations through the `sem:hasPlace` property. A `sem:Core` instance can be assigned an existence time denoted via `sem:hasBeginTimeStamp` and `sem:hasEndTimeStamp`. In addition to the SEM representation, *EventKG* provides textual information regarding events and entities extracted from the reference sources including labels (`rdfs:label`), aliases (`dcterms:alternative`) and descriptions of events (`dcterms:description`).

In the context of this thesis, the term *temporal relation* refers to real-world relations between events and entities valid over a period of time. The set of temporal relations in *EventKG* includes event-entity, entity-event and entity-entity relations. Temporal relations between events and entities typically connect an event and its actors (as in SEM), for example the marriage between two entities. Temporal relations

[3]Note that entities in *EventKG* are not necessarily actors in the events; temporal relations between two entities are also possible.

Table 3.4. Namespaces used in the *EventKG* RDF model.

Namespace prefix	IRI
so:	http://schema.org/
dbo:	http://dbpedia.org/ontology/
rdf:	http://www.w3.org/1999/02/22-rdf-syntax-ns#
rdfs:	http://www.w3.org/2000/01/rdf-schema#
dcterms:	http://purl.org/dc/terms/rdfs:
sem:	http://semanticweb.cs.vu.nl/2009/11/sem/
eventKG-s:	http://eventKG.l3s.uni-hannover.de/schema/
eventKG-r:	http://eventKG.l3s.uni-hannover.de/resource/
eventKG-g:	http://eventKG.l3s.uni-hannover.de/graph/

between entities can also indirectly capture information about events [RvEV+16]. For example, the DBpedia property `http://dbpedia.org/property/acquired` can be used to represent an event of acquisition of one company by another. Temporal relations in SEM are limited to the situation where an actor plays a specific role in the context of an event. This yields two limitations: (i) there is no possibility to model temporal relations between events and entities where the entity acts as a subject. For example, it is not possible to directly model the fact that Barack Obama participated in the event "Second inauguration of Barack Obama", as the entity "Barack Obama" plays the subject role in this relation; and (ii) a temporal relation between two entities such as a marriage can not be modelled directly[4].

To overcome these limitations, *EventKG* introduces the class `eventKG-s:Relation` representing relations between events and entities. This way of relation modelling facilitates additional flexible attributes describing a relation[5]. This class links two `sem:Core` instances (each representing an event or an entity). The resulting relation can be annotated with a validity time and a property `sem:RoleType` that characterises the relation using RDF predicates. Currently, the predicates are directly derived from the reference sources. In future work, we envision the normalisation of these predicates by mapping them to a dedicated ontology (e.g., the DBpedia ontology). This way, arbitrary temporal relations between entity pairs or relations involving an entity and an event can be represented. This model provides flexibility to express heterogeneous temporal relations derived from the reference sources. Figure 3.2 visualises the example mentioned above using the *EventKG* data model.

Other event and entity relations: Relations between events (in particular sub-event, previous and next event relations) play an important role in the context of event series (e.g., Olympic Games), seasons containing a number of related events (e.g., in sports), or events related to a certain topic (e.g., operations in a military conflict). Sub-event

[4]Consider the difference between a wedding that is modelled as an event and a marriage between two people that can be modelled as a temporal relation.

[5]See W3C Working Group Note from April 12, 2006 on defining N-ary Relations on the Semantic Web: https://www.w3.org/TR/swbp-n-aryRelations.

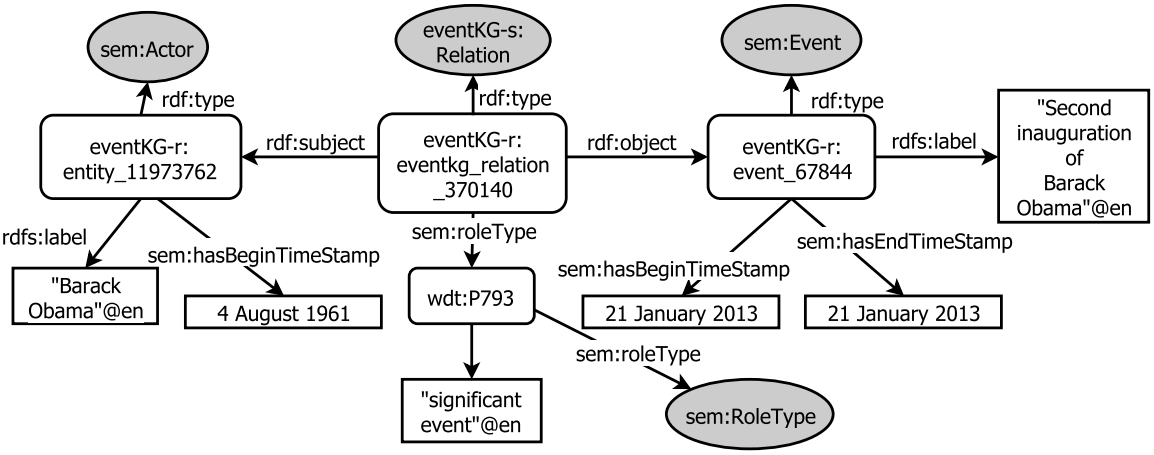


Figure 3.2. Example of the event representing the participation of Barack Obama in his second inauguration as a US president in 2013 as modelled in *EventKG*. `wdt:P793` is the Wikidata identifier for the "significant event" property.

relations are modelled using the `sem:hasSubEvent` property. To interlink events within an event series such as the sequence of the Olympic Games, the properties `dbo:previousEvent` and `dbo:nextEvent` are used. A location hierarchy is provided through the property `so:containedInPlace`.

Towards measuring relation strength and event popularity: Measuring relation strength between events and entities and event popularity enables answering questions like *"Who were the most important participants of the US Election 2016?"* or *"What are the most popular events related to the Summer Olympics 2016?"*. Relation strength and event popularity are of importance for many practical applications. For example, relation strength can help when using the knowledge graph to jointly disambiguate entities and events in text documents or natural language questions in the context of question answering applications. Relation strength and event popularity can also support ranking-based applications, including timeline generation and event-centric information retrieval.

Whereas the exact computation of relation strength and event popularity metrics can be application-dependent, we include two major factors required for such computations, namely *links* and *mentions* in the *EventKG* schema:

- *Links:* This factor represents how often the description of one entity refers to another entity. Intuitively, this factor can be used to estimate the popularity of events and the strength of their relations. In *EventKG*, the links factor is represented through the predicate `eventKG-s:links` in the domain of `eventKG-s:Relation`. `eventKG-s:links` denotes how often the Wikipedia article representing the relation subject links to the entity representing the object.
- *Mentions:* `eventKG-s:mentions` represents the number of relation mentions in

external sources. Intuitively, this factor can be used to estimate the relation strength. In *EventKG*, `eventKG-s:mentions` denotes the number of sentences in Wikipedia that mention both, the subject and the object of the relation.

Links and mentions factors provided by *EventKG* are computed using sources external to the knowledge graph, such as the entire Wikipedia corpus. Having this information included directly in the knowledge graph can help the relevant applications to obtain this information efficiently and to directly use it in their computations, including (but potentially not limited to) relation strength and event popularity metrics.

Provenance information: *EventKG* provides the following provenance information: (i) provenance of the individual resources; (ii) representation of the reference sources; and (iii) provenance of statements.

Provenance of the individual resources: *EventKG* resources typically directly correspond to the events and entities contained in the reference sources (e.g., an entity representing Barack Obama in *EventKG* corresponds to the DBpedia resource `http://dbpedia.org/resource/Barack_Obama`). In this case, the `owl:sameAs` property is used to interlink both resources. *EventKG* resources can also be extracted from a resource collection. For example, philosophy events in 2007 can be extracted from the Wikipedia event list at `https://en.wikipedia.org/wiki/2007_in_philosophy`. In this case, the *EventKG* property `eventKG-s:extractedFrom` is utilised to establish the link between the *EventKG* resource and the resource collection from which this resource was extracted. Through the provenance URIs, background knowledge contained in the reference sources can be accessed.

Representation of the reference sources: *EventKG* and each of the reference sources are represented through an instance of `void:Dataset`[6]. Such an instance in the namespace `eventKG-g` includes specific properties of the source (e.g., its creation date as in: `eventKG-g:dbpedia_pt dcterms:created "2016-10-01"^^xsd:date`).

Provenance information of statements: A statement in *EventKG* is represented as a quadruple, containing a triple and a URI of the named graph it belongs to. Through named graphs, *EventKG* offers an intuitive way to retrieve information extracted from the individual reference sources using SPARQL queries.

3.5.2 *EventKG* as a Temporal Knowledge Graph

A named graph such as `eventKG-g:event_kg` can be expressed as a temporal knowledge graph $G = (E_t, R_t)$ as follows:

- *Entities and events*: Each instance of `sem:Core` is a temporal entity $e \in E_t$ and each instance of `sem:Event` is an event $v \in \mathcal{V}$, such that $E = E_t \setminus \mathcal{V}$ is the set representing real-world entities.

[6] https://www.w3.org/TR/void/

- *Time information for entities and events*: For each temporal entity $e = \langle e_{uri}, e_{time} \rangle, e \in E_t$, e_{uri} is the URI of the corresponding *EventKG* entity. e_{start} and e_{end} are set according to the `sem:hasBeginTimeStamp` and `sem:hasEndTimeStamp` values in the `eventKG-g:event_kg` named graph, correspondingly.

- *Temporal relations with known validity times*: Each instance of `eventKG-s:Relation` that has a start or an end time in the named graph is transformed into a temporal relation $r = \langle r_{uri}, r_{time}, e_i, e_j \rangle \in R_t$. Here, r_{uri} is the URI of the *EventKG* relation instance, e_i is the entity connected to the `eventKG-s:Relation` instance via `rdf:subject`, e_j is the entity connected via `rdf:object` and r_{time} includes the `sem:hasBeginTimeStamp` and `sem:hasEndTimeStamp` relations.

- *Indirect temporal relations*: Information regarding the temporal validity of a relation is not always explicitly provided in *EventKG*. However, this information can often be derived based on the existence times of the participating entities or the happening times of the events. For example, the validity of a "mother" relation can be determined using the birth date of the child entity. We refer to such relations as *indirect temporal relations*. Each instance of `eventKG-s:Relation` that represents such an indirect temporal relation is transformed into a temporal relation $r_t = \langle r_{uri}, r_{time}, e_i, e_j \rangle \in R_t$, $r_{time} = e_{j_{time}}$.

3.5.3 *EventKG* Generation Pipeline

The *EventKG* generation pipeline is shown in Figure 3.3.

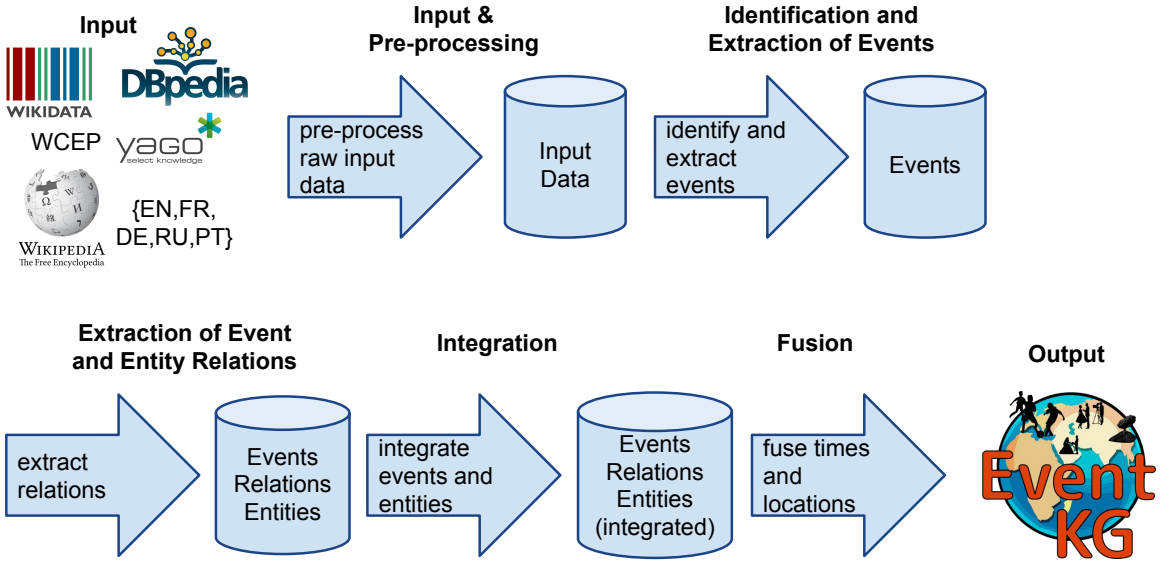


Figure 3.3. The *EventKG* generation pipeline.

Input & Pre-processing: First, the dumps of the reference sources in the corresponding languages are collected. Both Wikidata and YAGO provide multilingual information in a single data dump. DBpedia and Wikipedia provide language-specific dumps, so we collect the dumps for the languages of interest, i.e., English (EN), French (FR), German (DE), Russian (RU) and Portuguese (PT). The Wikipedia Current Events Portal is currently available in English only. The mapping from the Wikidata identifiers to the Wikipedia and DBpedia identifiers required for the integration is collected as part of the Wikidata dump.

As part of the pre-processing, the following information is created for each language[7]:

- *Terms:* Terms is a set of terms and regular expressions used throughout the extraction process. This includes the month names, weekday names, a blacklist of namespaces and prefixes of the Wikipedia articles to be ignored (e.g., the prefix "Chronological_list_of_" in English) as well as regular expressions to detect titles of the Wikipedia articles representing events.

- *Date expressions:* To extract dates from unstructured reference sources, a set of regular expressions is created. These expressions are sorted in the decreasing order of specificity, where time intervals are considered to be more specific than the individual dates or months. For example, a specific regular expression to extract a span of two dates in English is: `@regexMonthDay1@@hyphensOr@@regexMonthDay2@`, where `@regexMonthDay1@` denotes a month name followed by a date and `@hyphensOr@` is any kind of hyphen. This regular expression can match textual patterns such as "February 17 — April 23". A less specific expression is `@regexDay1@` that only checks for day numbers such as "17". Moreover, regular patterns to identify Wikipedia event lists such as "2007 in Science" are created, together with the rules to extract the temporal scope (the year 2007 in this example).

- *Mapping of predicates representing event relations:* We define a mapping table to identify predicates that represent equivalent event relations in *EventKG* and its reference sources such as `so:hasSubEvent` and Wikidata's "`part of`" property. Examples of such mappings are shown in Table 3.5. Currently, we define the predicate mappings manually. In future work, schema mapping techniques can be adopted to determine such links automatically.

Identification and Extraction of Events: Event instances are identified in the reference sources and extracted as follows:

[7]To obtain a complete list of the manually defined terms, expressions and mappings adopted in this work, please see the readme file in the open source software release provided at: https://github.com/sgottsch/eventkg

- **Wikidata** [EGK+14]: We identify events as subclasses of Wikidata's "event" (representing temporary and scheduled events like festivals or competitions) and "occurrence" (representing happenings like wars or ceremonies). Some of the identified subclasses are blacklisted manually. For example, the class "song" is blacklisted because of the subclass hierarchy `song` > `musical form` > `art form` > `format` > `arrangement` > `act` > `process` > `occurrence`.

- **DBpedia** [LIJ+15]: For each language edition, we identify DBpedia events as instances of `dbo:Event` or its subclasses.

- **YAGO** [MBS14]: We do not use the YAGO ontology for event identification due to the noisy event subcategories we observed (e.g., `event` > `act` > `activity` > `protection` > `self-defense` > `martial_art`).

- **Wikipedia**: We use Wikipedia category names that match a manually defined language-dependent regular expression (e.g., English category names that end with " events") as an indication that a knowledge graph entry linked to such an article is an event.

- **Wikipedia Event Lists**: For each language, we identify Wikipedia event lists by adopting a set of regular expressions defined manually during pre-processing. This way, Wikipedia pages with titles such as "2007 in Science" and "August 11" are retrieved. Within these pages, textual descriptions of events are collected using methods similar to [HWP12]. Using the ordered list of regular temporal expressions and Wikipedia link markup, representations of events including their descriptions, linked entities and dates are extracted.

- **WCEP**: In the Wikipedia Current Events Portal, events are represented through rather brief textual descriptions and refer to daily happenings. We extract WCEP events using the WikiTimes tool [TA14].

Extraction of Event and Entity Relations: We extract the following types of relations:

1. *Relations with temporal validity* are identified based on the availability of temporal information. Temporal relations are extracted from YAGO and Wikidata. DBpedia does not provide such information.

2. *Relations with indirect temporal information*: We extract all relations involving events as well as relations of entities with known existence time.

3. *Other event and entity relations*: We use the manually defined mapping table shown in Table 3.5 to identify predicates that represent event relations in *EventKG* such as `so:hasSubEvent` (e.g., we map Wikidata's `part of` property (P361) to `so:hasSubEvent` in cases where the property is used to connect events),

Table 3.5. Example property mapping between *EventKG* and its reference sources.

EventKG	**Wikidata**	**DBpedia**	**YAGO**
sem:hasPlace	wd:P276 (location) wd:P30 (continent) ...	dbo:place	yago:isLocatedIn yago:happenedIn
sem:hasBegin-TimeStamp	wd:P580 (start time) wd:P585 (point in time) wd:P1619 (date of off. opening) ...	—	yago:started-OnDate yago:happenedOn-Date
sem:hasEnd-TimeStamp	wd:P582 (end time) wd:P585 (point in time) ...	—	yago:endedOnDate yago:happenedOn-Date
sem:hasSub-Event	wd:P361 (part of)	dbo:isPartOf dbo:isPartOf-MilitaryConflict ...	—
so:previous-Event	wd:P155 (follows)	dbo:previousEvent dbo:previousWork	—
so:nextEvent	wd:P156 (followed by)	dbo:followingEvent dbo:subsequentWork	—
so:contained-InPlace	wd:P36 (capital) wd:P706 (loc. on terrain feat.) ...	—	—

dbo:previousEvent and dbo:nextEvent as well as so:containedInPlace to extract location hierarchies.

4. *Relation strength and event popularity information*: For each event-entity relation, we extract language-specific interlinking information from Wikipedia. In particular, we extract the number of links and the number of mentions for each relation involving events. Link and mentions are extracted from each Wikipedia language edition by parsing all of its pages.

Integration: The statements extracted from the reference sources are included in the named graphs, such that each named graph corresponds to a reference source. In addition, we create a named graph eventKG-g:event_kg containing information resulting from integration and fusion. Each sem:Event and sem:Core instance in the eventKG-g:event_kg graph integrates event-centric and entity-centric information from the reference sources related to equivalent real-world instances.

The integration of entities and events obtained from knowledge graphs and Wikipedia articles is conducted using existing owl:sameAs links, as provided by the Wikidata dataset. In particular, the entities and events covered by YAGO and different language

versions of DBpedia and Wikipedia are also present in Wikidata. We use `owl:sameAs` links to the Wikidata identifiers to represent each resource that is linked as equivalent in multiple reference sources as one resource in *EventKG*. That way, information regarding this resource in different reference sources, e.g., labels in different languages, is integrated.

The events in the Wikipedia event lists and WCEP do not possess unique identifiers. Such events are integrated using a rule-based approach to identify equivalent events. Two events e_1 and e_2 extracted from such sources are represented as one *EventKG* event if the times of these events are identical ($e_1.time = e_2.time$) and the set of entities they link to overlaps. A special case is given if an event e_1 without an identifier links to exactly one event e_n with a known identifier and their times are equal. In that case, the text of e_1 is added as a description to e_n.

Fusion: In the fusion step, we aggregate temporal, spatial and type information of `eventKG-g:event_kg` events using a rule-based approach.

- *Time Fusion*: For each entity, event or relation with a known existence or a validity time stamp, time fusion is conducted using the following rules: (i) ignore the dates at the beginning or end of a time unit (e.g., January, 1st), if alternative dates are available; (ii) apply majority voting among the reference sources; (iii) take the time stamp from the more trusted source (in order: Wikidata, DBpedia, Wikipedia, WCEP, YAGO).
- *Location Fusion*: For each event in `eventKG-g:event_kg`, we take the union of its locations from the different reference sources and exploit the `so:containedInPlace` relations to reduce this set to the minimum (e.g., the set {Paris, France, Lyon} is reduced to {Paris, Lyon}, while France can still be induced as a location using `so:containedInPlace` transitively).
- *Type Fusion*: We provide `rdf:type` information according to the DBpedia ontology (`dbo`), using types and `owl:sameAs` links in the reference sources.

Output: Finally, extracted instances and relations are represented in RDF according to the *EventKG* data model (see Section 3.5.1). As described above, information extracted from each reference source and the results of the fusion step are provided in separate named graphs.

3.5.4 Running Example: Barack Obama

In the context of our running example, we now provide an exemplary overview of the *EventKG* generation pipeline and illustrate how exemplar relations are expressed in the *EventKG* model and the temporal knowledge graph. We refer to individual heterogeneous instances in the input data that are not yet expressed in the *EventKG* schema as *data items*. Table 3.6 provides exemplary data items involving Barack

Obama obtained from Wikidata, YAGO and different language editions of Wikipedia and DBpedia.

Table 3.6. Example data items about Barack Obama extracted from different reference sources.

#	Reference Source	Data Item	Related Data Items
1	Wikipedia$_{EN}$	*May 8, 2018*: President Trump announces his intention to withdraw the United States from the Iranian nuclear agreement. In a statement, former U.S. President Barack Obama calls the move "a serious mistake".	—
2	Wikidata	*Barack Obama*, *significant event*, *first inauguration of Barack Obama*	Wikidata: *first inauguration of Barack Obama*, *point in time*, *20 January 2009* YAGO: *first inauguration of Barack Obama*, *was created on*, *17 July 1981* Wikidata: *first inauguration of Barack Obama*, *instance of*, *United States presidential inauguration* Wikidata: *United States presidential inauguration*, *subclass of**, *occurrence*
3	Wikidata	*Barack Obama*, *spouse*, *Michelle Obama - start time*: *3 October 1992*	—
4	DBpedia$_{FR}$	*Barack Obama*, *prop-fr:candidat*, *Élection présidentielle américaine de 2012*	DBpedia$_{FR}$: *Élection présidentielle américaine de 2012 owl:sameAs United States presidential election, 2012* Wikidata: *United States presidential election, 2012*, *point in time*, *6 November 2012*
5	Wikipedia$_{PT}$	*[The Portuguese Wikipedia page of Barack Obama links to the page "Death of Osama bin Laden" once.]*	Wikidata: *Death of Osama bin Laden*, point in time, *2 May 2011*

Identification and Extraction of Events. The first data item is extracted from the English Wikipedia event list in the article "2018 in the United States". The entities "first inauguration of Barack Obama", "United States presidential election, 2012" and "Death of Osama bin Laden" from the data items #2, #3 and #5 are identified as events using the class hierarchies in the reference sources. In this example, Obama's first inauguration is identified as an event, because it is an instance of "United States presidential inauguration", which can be traced back to `inauguration` > `key event` > `occurrence` in Wikidata. Thus, the text event from data item #1 and the event "first inauguration of Barack Obama" are stored as event instances with additional

values such as a textual description for the former and a title for the latter event.

Extraction of Event and Entity Relations. Given the set of events, we can now detect relations between them and other entities. For example, the statement that Barack Obama was involved in his own inauguration as US president is extracted from Wikidata. This statement represents an indirect temporal relation, as it alone does not provide the required temporal validity information, which needs to be extracted from a related fact about the event. Similarly, we can extract the information that Barack Obama was a candidate of the US elections in 2012 from the French DBpedia.

With the help of Wikipedia links, we connect Barack Obama to the death of Osama bin Laden (data item #5). Given the relation `?rel` that links to Barack Obama as the subject and to the event "Death of Osama bin Laden" as the object, the link information is modelled as follows, using a named graph (where `eventKG-r:entity_11973762` represents Barack Obama and `eventKG-r:event_527087` represents the event "Death of Osama bin Laden"):

```
?rel rdfs:type
      eventKG-s:Relation .
?rel rdf:subject
      eventKG-r:entity_11973762 .
?rel rdf:object
      eventKG-r:event_527087 .

eventKG-g:wikipedia_pt {
  ?rel eventKG-s:links 1 .
} .
```

For the relation `?rel`, link information can be added using specific named graphs. For example, such information can model the co-mentions of Barack Obama and the death of Osama bin Laden in the Portuguese Wikipedia.

Another type of information is coming from the temporal relations between two temporal entities: Here, the *spouse* relation between Barack and Michelle Obama is directly assigned a temporal validity time by Wikidata.

Integration. The entities "Èlection présidentielle américaine de 2012" and "United States presidential election, 2012," are modelled as the same event resource in *EventKG*, using DBpedia's `owl:sameAs` link.

Fusion. There are two different dates provided for the first inauguration of Barack Obama (data item #2). While both dates are stored in *EventKG* together with their provenance information (i.e., as named graphs for Wikidata and YAGO), a single happening time for that event is created with our rule-based fusion approach (see Section 3.5.3). As the majority voting is not sufficient here, we take the date from the higher trusted source. In this case, Wikidata's date (January 20, 2009) is selected for *EventKG*'s named graph.

With that time information, the indirect temporal relation about Obama's participation in his own inauguration can be transformed into the following temporal relation in the temporal knowledge graph generated from the named graph `eventKG-g:event_kg`:

```
Barack Obama,
significant event:
first inauguration of Barack Obama
[2009-01-20,2009-01-20]
```

3.6 Evaluation & Statistics

To demonstrate the quality of the data extraction, integration and fusion steps, we first show characteristics of *EventKG* and provide several comparisons to its reference sources in Section 3.6.1. Then, we provide evaluation results based on user annotations in Section 3.6.2.

3.6.1 Characteristics

In *EventKG* V1.1, we extracted event representations and relations in five languages – English (EN), German (DE), French (FR), Russian (RU) and Portuguese (PT) – from the latest available versions of each reference source as of 12/2017. *EventKG* uses open standards and is publicly available under a persistent URI[8] under the CC BY 4.0 license[9]. Our extraction pipeline is available as open-source software on GitHub[10] under the MIT License[11]. A description of *EventKG* and example SPARQL queries are online[12].

Table 3.7 summarises selected statistics from *EventKG* V1.1, released in 03/2018. Overall, this version provides information for over 690 thousand events and over 2.3 million temporal relations. Nearly half of the events (46.75%) originate from the existing knowledge graphs; the other half (53.25%) is extracted from semi-structured sources. The data quality of the individual named graphs directly corresponds to the quality of the reference sources. In `eventKG-g:event_kg`, the majority of the events (76.21%) possess a known start or end time. Locations are provided for 12.21% of the events. The coverage of locations can be further increased in future work, e.g., using NLP techniques to extract locations from event descriptions.

Along with over 2.3 million temporal relations, *EventKG* V1.1 includes relations between events and entities for which the time is not available. This results in overall

[8] https://doi.org/10.5281/zenodo.1112283
[9] https://creativecommons.org/licenses/by/4.0/
[10] https://github.com/sgottsch/eventkg
[11] https://opensource.org/licenses/MIT
[12] http://eventkg.l3s.uni-hannover.de/

Table 3.7. Number of events and relations in `eventKG-g:event_kg`.

	#Events	Known time	Known location
Events from KGs	322,669	163,977	84,304
Events from semi-structured sources	367,578	362,064	not extracted
Relations	88,473,111	2,331,370	not extracted

Table 3.8. Number of events identified and extracted from the reference sources.

	DBpedia					Wikipedia event lists					
Wikidata	**EN**	**FR**	**DE**	**RU**	**PT**	**EN**	**FR**	**DE**	**RU**	**PT**	**WCEP**
266,198	60,307	43,495	9,383	5,730	14,641	131,774	110,879	21,191	44,025	18,792	61,382

Table 3.9. Comparison of the event representation completeness in the source-specific named graphs (after integration).

				DBpedia				
	EventKG	**Wikidata**	**YAGO**	**EN**	**FR**	**DE**	**RU**	**PT**
#Events with	322,669	322,669	222,325	214,556	78,527	62,971	47,304	35,682
Location (L)	26.13%	11.70%	26.61%	6.21%	8.32%	4.03%	10.60%	6.15%
Time (T)	50.82%	33.00%	39.02%	7.00%	17.21%	2.00%	1.35%	0.08%
L&T	21.97%	8.83%	19.02 %	4.29%	0.00%	4.84%	1.18%	0.08%

over 88 million relations. Approximately half of these relations possess interlinking information.

Comparison of *EventKG* to its Reference Sources

We compare *EventKG* to its reference sources in terms of the number of identified events and completeness of their representations. The results of the event identification and extraction step in Section 3.5.3 are shown in Table 3.8. *EventKG* V1.1 with 690,247 events contains a significantly higher number of events than any of its reference sources, which comes from the integration of knowledge graphs and semi-structured sources.

Table 3.9 presents a comparison of the event representations in *EventKG* and its reference knowledge graphs (Wikidata, YAGO, DBpedia). As we can observe, through the integration of event-centric information, *EventKG*: 1) enables better event identification (e.g., we can map 322,669 events from *EventKG* to Wikidata, whereas only 266,198 were identified as events in Wikidata initially - see Table 3.8) and 2) provides more complete event representations (i.e., *EventKG* provides a higher percentage of events with specified temporal and spatial information compared to Wikidata, that is the most reference source which covers most events). The most frequent event types are source-dependent (see Table 3.10).

Table 3.10. The most frequent event types extracted from the references sources and the percentage of the events in that source with the respective type.

	Wikidata	DBpedia EN	DBpedia FR	DBpedia DE	DBpedia RU	DBpedia PT
dbo:type	season	Military Conflict	Sports Event	Tennis Tournament	Military Conflict	Soccer Tournament
Events, %	11.37%	6.31%	21.86%	33.00%	11.87%	16.17%

Relation & Fusion Statistics

More than 2.3 million temporal relations are an essential part of *EventKG*. The majority of the frequent predicates in *EventKG* such as "member of sports team" (882,398 relations), "heritage designation" (221,472), "award received" (128,125) and "position held" (105,333) originate from Wikidata. The biggest fraction of YAGO's temporal relations has the predicate "plays for" (492,263), referring to football players. Other YAGO predicates such as "has won prize" are less frequent. Overall, about 93.62% of the temporal relations have a start time from 1900 to 2020. 81.75% of events extracted from knowledge graphs are covered by multiple sources. At the fusion step, we observed that 93.79% of the events that have a known start time agree on the start times across the different sources.

Textual Descriptions

EventKG V1.1 contains information in five languages. Overall, 87.65% of the events extracted from knowledge graphs provide an English label, whereas only a small fraction (4.49%) provides labels in all languages. Among the 367,578 events extracted from the semi-structured sources, just 115 provide a description in all five languages, e.g., the first launch of a Space Shuttle in 1981. This indicates the potential for further enrichment of multilingual event descriptions in future work.

3.6.2 Evaluation of *EventKG*

The aim of the evaluation is to assess the effectiveness of the event identification, time fusion and location fusion steps of the pipeline.

Event Identification

We manually evaluated a random sample of the events identified in the event identification step of *EventKG* (Section 3.5.3). For each reference source, we randomly sampled 100 events and manually annotated whether they represent real-world events or not. The results are shown in Table 3.11.

Table 3.11. User-evaluated precision for the identification of events with selected reference sources.

		DBpedia			Wikipedia	
	Wikidata	**DE**	**RU**	**PT**	**EN**	**RU**
Precision	96%	100%	100%	98%	94%	88%

Table 3.12. Evaluation of *EventKG*'s time information. For *EventKG* and the reference sources, the percentage of correct, wrong and missing event dates with respect to the user annotations in our sample is shown. These are based on the random sample of events where the reference sources show disagreement between time information provided (Corr.: Correct, Prec.: Precision).

	Start Dates			End Dates			Start and End Dates			
Source	**Corr.**	**Wrong**	**Missing**	**Corr.**	**Wrong**	**Missing**	**Corr.**	**Wrong**	**Missing**	**Prec.**
EventKG	**71**	25	0	**73**	23	0	**144**	48	0	**0.75**
Wikidata	40	33	23	33	29	34	73	62	57	0.54
YAGO	21	60	15	20	57	19	41	117	34	0.26
$\text{DBpedia}_{\text{EN}}$	12	5	79	13	4	79	25	9	158	0.74
$\text{DBpedia}_{\text{DE}}$	0	2	94	2	0	94	2	2	188	0.5
$\text{DBpedia}_{\text{FR}}$	6	17	73	15	8	73	21	25	146	0.46
$\text{DBpedia}_{\text{RU}}$	0	2	94	0	2	94	0	4	188	0

For DBpedia and Wikidata, where we rely on the event types and type hierarchies, we achieve a precision of 98% on average. On a random sample of 100 events extracted from the category names in the English and the Russian Wikipedia, we achieve 94% and 88% precision, correspondingly. One example for an entity wrongly identified as an event is the cancelled project "San Francisco Municipal Wireless", which was part of the "Cancelled projects and events" category in Wikipedia.

Time Fusion

To evaluate the quality of the proposed rule-based time fusion approach, we randomly sampled 100 events from *EventKG*, where each event has at least two reference sources that differ in the event happening time (i.e., start and/or end time). Three users have annotated this sample by providing a start and end time for at least 20 events each. Additionally, we asked the users to denote which source they used to research the actual event dates. For our evaluation, we then checked how many of the user-given start and end dates are available in the reference sources and the joint *EventKG* named graph, and we computed how many of these dates are correct with respect to the user annotations.

Table 3.12 provides the result overview: As the time fusion does always adopt accessible time information from any reference source, all events in our random sample

Table 3.13. Time fusion evaluation: The most frequent sources used by the users to lookup event start and end dates.

Source	#Uses	Percentage
en.wikipedia.org	117	58.5%
www.google.com	37	18.5%
de.wikipedia.org	14	7.0%
no source used	7	3.5%
fr.wikipedia.org	6	3.0%
www.singapore-elections.com	2	1.0%
www.un.org	2	1.0%
...		

possess time information. Wikidata and YAGO provide the next highest coverage of time information. In terms of precision, *EventKG* outperforms these two reference sources by 21% (Wikidata) and 49% (YAGO). This result confirms the quality of the proposed rule-based time fusion approach.

The results of a McNemar's test [McN47] has shown a two-tailed p-value of less than 0.0001, which confirms the statistical significance of this result.

Table 3.13 provides an overview of the sources most often used for finding the event dates by the users participating in the evaluation. In 69% of the cases, the users adopted Wikipedia articles in different languages as their source. When the users did not use Wikipedia, either the information presented on the search engine's result page (18.5% of the cases) or domain-specific web sites such as www.singapore-elections.com or www.un.org were used.

Location Fusion

To evaluate the correctness of the extracted locations, we selected a random sample of 100 events with at least one location. In the case of locations, multiple correct values are possible; for example, South America, the United States of Colombia and the Colombia-Ecuador border are valid locations for the Ecuadorian-Colombian War. We presented all locations from each reference source to the users and for each location asked the users to verify whether that location is correct or not. Four users have annotated that sample.

Table 3.14 provides the result for our evaluation of the location fusion. We distinguish between the locations directly provided by *EventKG* and those which could be inferred using sub-location information via `so:containedInPlace`. We refer to this extended knowledge graph as *EventKG** throughout this evaluation. *EventKG* and *EventKG** have by far the highest coverage of locations (*EventKG** finds 78.13% more event locations than YAGO and 159.10% more than in Wikidata), while keeping

Table 3.14. Evaluation of *EventKG*'s location information. For each event in the sample, users judged for each location in *EventKG* and the reference sources whether it is correct.

Source	Correct	Wrong	Precision
*EventKG**	**116**	7	94.31%
EventKG	87	4	95.60%
YAGO	64	2	96.97%
Wikidata	44	2	95.65%
$\text{DBpedia}_{\text{EN}}$	15	1	93.75%
$\text{DBpedia}_{\text{FR}}$	7	0	100.0%
$\text{DBpedia}_{\text{DE}}$	1	0	100.0%
$\text{DBpedia}_{\text{RU}}$	4	1	80.0%
$\text{DBpedia}_{\text{PT}}$	3	1	75.0%

Table 3.15. Location fusion evaluation: The most frequent sources used by the users to lookup event locations.

Source	#Uses	Percentage
en.wikipedia.org	58	43.94%
no source used	35	26.51%
de.wikipedia.org	7	5.3%
www.google.com	5	3.79%
everipedia.org	3	2.0 %
fr.wikipedia.org	3	2.0 %
www.kicker.de	2	1.51%
	...	

the number of wrong locations low (approx. 7%). However, *EventKG* also inherits wrong locations as provided by the reference sources due to the adopted location fusion mechanism.

The results of a McNemar's test [McN47] has shown a two-tailed p-value of 0.0005, which confirms statistical the significance of this result.

Table 3.15 lists the sources used by the users in this task. Similarly to the evaluation of the time fusion, Wikipedia and Google were the most frequently used sources, followed by domain-dependent ones such as kicker.de for locating football matches. However, in 26.51% of the cases in this task, the users did not use a source at all, mainly because many event locations are self-explanatory or contained in the event names. For example, no source was needed to verify the locations Monaco and Circuit de Monaco for the 1956 Monaco Grand Prix.

3.6.3 *EventKG* V3.0

The characteristics, statistics and evaluation results presented in this chapter refer to *EventKG* V1.1 released in March 2018.

In March 2020, we released *EventKG* V3.0 that includes several updates[13] with respect to the: i) inclusion of the current content of the reference sources and extended language coverage, ii) enhanced relation fusion, iii) inclusion of geographic information, iv) inclusion of information regarding temporal granularity, v) inclusion of an event series type. In the following, we describe these extensions in more detail.

Reference sources and language coverage. *EventKG* V3.0 includes data extracted from the reference sources presented in Section 3.5.3 as of February 20th, 2020. Furthermore, *EventKG* V3.0 includes ten more languages, in addition to the five languages supported in *EventKG* V1.1: Italian, Spanish, Dutch, Polish, Norwegian, Romanian, Croatian, Slovene, Bulgarian and Danish. Overall, this leads to 1,348,561 events included in the dataset.

Relation fusion. In *EventKG* V3.0, we performed fusion of `eventKG-s:Relation` instances extracted from different reference sources based on property mappings and similarity. `eventKG-s:Relation` instances are fused if the following conditions are met: (1) The values of `rdf:subject`, `rdf:object`, `sem:hasBeginTimeStamp` and `sem:hasEndTimeStamp` are the same, and (2) the `sem:roleType` values are linked via existing `owl:sameAs` relations in the reference sources. For example, this concerns properties such as "place of birth" (Wikidata), "wasBornIn" (YAGO) and "birthPlace" (English DBpedia).

Geographic information. For `sem:Place` and `sem:Event` instances, geographic coordinates available in the reference sources are added to *EventKG* V3.0. The coordinates are represented through their latitude and longitude as values of `so:latitude` and `so:longitude`.

Temporal granularity information. In *EventKG* V3.0, we enriched the dates encoded by `sem:hasBeginTimeStamp` and `sem:hasEndTimeStamp` with granularity information, which denotes the precision of a given date. To this end, the properties `eventKG-s:startUnitType` and `eventKG-s:endUnitType` are added to the schema. Their range is `time:TemporalUnit`, which comprises existing classes in the Time Ontology[14] (`time:unitDay`, `time:unitMonth` and `time:unitYear`), as well as newly created classes (`eventKG-s:unitDecade` and `eventKG-s:unitCentury`). For example, the granularity information helps to identify whether the start time "January 1st, 1981" refers to that actual day (`eventKG-s:startUnitType time:unitDay`) or to an unknown day of the year (`eventKG-s:startUnitType time:unitYear`).

Event series. Two new classes were added to the schema of *EventKG*: `eventKG-s:EventSeries` and `eventKG-s:EventSeriesEdition`, which represent event series

[13] Here, we list updates from *EventKG* V1.1 to *EventKG* V3.0, including the updates of *EventKG* V2.0.

[14] http://www.w3.org/2006/time# (namespace prefix "`time:`")

(such as the Wimbledon Championships) and their editions (e.g., the Wimbledon Championships in 2018).

EventKG V3.0, its updated schema information and statistics, are accessible online.[15]

3.7 Discussion

In this chapter, we presented the concept of a temporal knowledge graph that interconnects real-world entities and events using temporal relations. Furthermore, we presented an instantiation of the temporal knowledge graph – *EventKG*. *EventKG* is a multilingual knowledge graph that integrates and harmonises event-centric and temporal information regarding historical and contemporary events. *EventKG* V1.1 includes over 690 thousand event resources and over 2.3 million temporal relations. Unique *EventKG* features include the light-weight integration and fusion of structured and semi-structured multilingual event representations and temporal relations in a single knowledge graph, as well as the provision of information to facilitate the assessment of relation strength and event popularity while providing provenance. The light-weight integration enables to significantly increase the coverage and completeness of the included event representations, in particular with respect to time and location information.

We analysed the characteristics of the resulting knowledge graph and observed a significant increase in coverage compared to the reference sources. For example, *EventKG* V1.1 contains 50K more events than identified in Wikidata and more than 262K events than identified in the English DBpedia. Additionally, 360K events are extracted from semi-structured sources. The quality of this resulting dataset was confirmed in a manual evaluation. This evaluation indicated high precision for the event identification step (with an average precision of 96%), the time fusion step (with precision of 75% for the events that had a disagreement regarding their time information in the reference sources) and the precision of the location fusion (94.31%).

The characteristics, statistics and evaluation results presented in this chapter refer to *EventKG* V1.1 released in March 2018. In March 2020, we released *EventKG* V3.0, briefly described in Section 3.6.3. In comparison to *EventKG* V1.1, *EventKG* V3.0 includes an increased number of more than 1.3 million events, further enhances relation fusion, provides geographical information, an event series type and *EventKG* V3.0 integrates reference sources in 15 languages in total.

EventKG demonstrates the importance of adapting knowledge graphs to the given setting: While well-established knowledge graphs such as Wikidata or DBpedia serve as an excellent foundation for a large variety of general-purpose applications, they may not be the perfect fit for other applications that are very domain-specific or have

[15] http://eventkg.l3s.uni-hannover.de/ and https://doi.org/10.5281/zenodo.1112283

specific demands for the knowledge they utilise. At this point, the integration of several, heterogeneous sources, is an important step towards the application of event knowledge graphs.

4 Creation of a Knowledge Graph from Tabular Data

Knowledge comes in many ways, not limited to existing knowledge graphs or encyclopedias. There are plenty of datasets around which are not understandable by computers. One example are tabular datasets, which are structured into rows and columns but lack semantic annotations. Typical data analytics tasks that aim at generating insights from such data tables, oftentimes using machine learning, can highly benefit from semantic annotations [GTK+19]. However, as several studies confirm [Moo18, Pre16], the organisation and pre-processing of data is a highly time-consuming task, albeit not as enjoyable as other tasks, including the configuration of a machine learning workflow. But what if we can automatically transform input data tables into a knowledge graph, and thus ease the whole process of data understanding, organisation and cleaning? Such transformations require semantic table interpretation. Following **RQ1.2**, this chapter deals with the problem of semantic table interpretation in a specific setting, where tables are interpreted based on background knowledge, represented using domain profiles. Consequently, this chapter is about *Tab2KG*, an approach for transforming tabular data into knowledge graphs using domain profiles.

4.1 Introduction

A vast amount of data is currently published in a tabular format [CHL+18, RB17, MNUP16]. Typically, this data does not possess any machine-readable semantics. Semantic table interpretation is an essential step to make this data usable for a wide variety of applications, with data analytics workflows (DAWs) as a prominent example [GTK+19]. DAWs include data mining algorithms and sophisticated deep learning architectures and require a large amount of heterogeneous data as an input. Typically, DAWs treat tabular data as character sequences and numbers without inferring any further semantics. This practice can often lead to error-prone analytics processes and results, particularly when data analytics frameworks utilize the data from various heterogeneous sources. Therefore, DAWs can substantially profit from the semantic

interpretation of the involved data tables [Gar19, PPK+18].

In this context, semantic table interpretation aims to transform the input data table into a semantic data graph. In this process, table columns are mapped to a domain ontology's classes and properties; table cell values are transformed into literals, forming the data graph – a network of semantic statements, typically encoded in RDF. This process is typically called *semantic table interpretation*, as introduced in Section 2.3.3.

In the context of DAWs, semantic table interpretation can bring several advantages. First, an abstraction from tabular data to semantic concepts and relations can guide domain experts in the DAW creation [GTK+19]. Second, validation options (e.g., type inference) that become available through the semantic table interpretation can increase the robustness of DAWs [HLLS17]. Third, semantic descriptions can be employed to facilitate the explainability of the data analytics results [Lec19]. Fourth, semantic table interpretation adds structure directly usable for knowledge inference [LSB+17].

The existing approaches to semantic table interpretation do not adequately support the semantic interpretation of tabular data for DAWs. At the core of such approaches (e.g., [NKIT19, CDPRS20, CJRHS19]) is the instance lookup task, where table cells are linked to known instances in a target knowledge graph, with DBpedia [ABK+07] being a popular cross-domain target. Subsequent steps such as property mapping are based on the results of this lookup step. However, as shown by Ritze et al. [RLOB16], only about 3% of the tables contained in the 3.5 billion HTML pages of the Common Crawl Web Corpus[1] can be matched to DBpedia. In the context of DAW, the input data typically represents new instances (e.g., sensor observations, current road traffic events, ...), and substantial overlap between the tabular data values and entities within existing knowledge graphs cannot be expected.

The unavailability of known instances becomes particularly evident in the context of DAWs, where input data typically represents new knowledge. Consider the following three exemplary DAW tasks:

- Weather prediction based on sensor observations that do not contain any references to instances in a knowledge graph.
- Traffic jam prediction based on a file of recent events not yet covered in the target knowledge graph.
- An analysis of recent events that are not yet present in cross-domain knowledge graphs such as DBpedia.

These examples demonstrate that approaches solely relying on matching table cells to resource in a knowledge graph are not applicable in common DAW settings.

[1] http://commoncrawl.org/

In this article, we present *Tab2KG* – a novel semantic table interpretation approach. *Tab2KG* aims to transform a data table into a semantic data graph. As a backbone of the data graph, *Tab2KG* relies on an existing domain ontology that defines the concepts and relations in the target domain. To facilitate the transformation, *Tab2KG* introduces original lightweight semantic profiles for domains and data tables. Domain profiles enrich ontology relations and represent domain characteristics. A domain profile associates relations with feature vectors representing data types and statistical characteristics such as value distributions. To transform a data table, *Tab2KG* first creates a data table profile. Then, *Tab2KG* uses the domain and data table profiles to transform the table into a data graph using a novel one-shot learning approach.

To create domain profiles, *Tab2KG* uses a domain ontology and a data sample that can be obtained from an existing domain knowledge graph. Lightweight semantic profiles generated by *Tab2KG* can be utilized as compact domain representations and complement and enrich existing dataset catalogs. Such profiles can be generated automatically from the existing datasets and described using the DCAT[2] and the SEAS[3] vocabularies to enable their reusability. We believe that lightweight semantic profiles presented in this article are an essential contribution that can benefit a wide range of semantic applications beyond semantic table interpretation.

4.1.1 Contributions

In summary, our contributions presented in this chapter are as follows:

1. We introduce lightweight semantic domain and table profiles. Domain profiles enrich relations of domain ontologies and serve as a lightweight domain representation. Semantic table profiles summarize data tables facilitating effective semantic table interpretation.

2. We propose the *Tab2KG* approach to transform tabular data into a data graph with one-shot learning based on semantic profiles.

3. We evaluate the proposed method on several real-world datasets. Our evaluation results demonstrate that *Tab2KG* outperforms state-of-the-art semantic table interpretation baselines.

4. We make the scripts for creating lightweight semantic profiles and transforming data tables into data graphs publicly available[4].

[2] https://www.w3.org/TR/vocab-dcat-2/
[3] https://ci.mines-stetienne.fr/seas/index.html
[4] https://github.com/sgottsch/Tab2KG

4.1.2 Running Example: Weather Observations

Tab2KG is a domain-independent approach that generalizes to previously unseen domains and data tables. There are no constraints on the data nature (e.g., sensor data, numbers, strings, ...), and we demonstrate in our evaluation how *Tab2KG* performs on different domains, including soccer and advertisements data. As a new running example throughout this chapter, we use the weather observation domain and a data table that provides observations of sensors measuring air conditions.

```
cloudy  ↠  16:30  ↠  17:00  ↠  S2
clear   ↠  10:00  ↠  10:30  ↠  S3
cloudy  ↠  17:00  ↠  17:30  ↠  S3
rain    ↠  16:30  ↠  17:00  ↠  S1
cloudy  ↠  17:30  ↠  18:00  ↠  S2
clear   ↠  08:30  ↠  09:00  ↠  S1
clear   ↠  09:00  ↠  09:30  ↠  S1
```

Figure 4.1. Example of a data table as a tab-separated file without column titles.

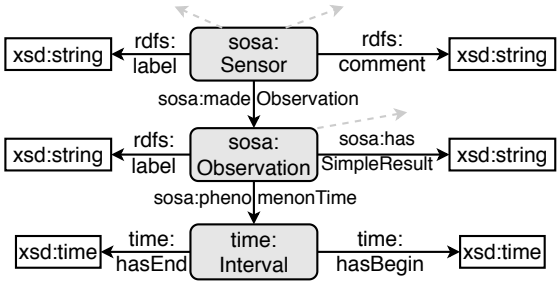


Figure 4.2. Excerpt of the Semantic Sensor Network Ontology.

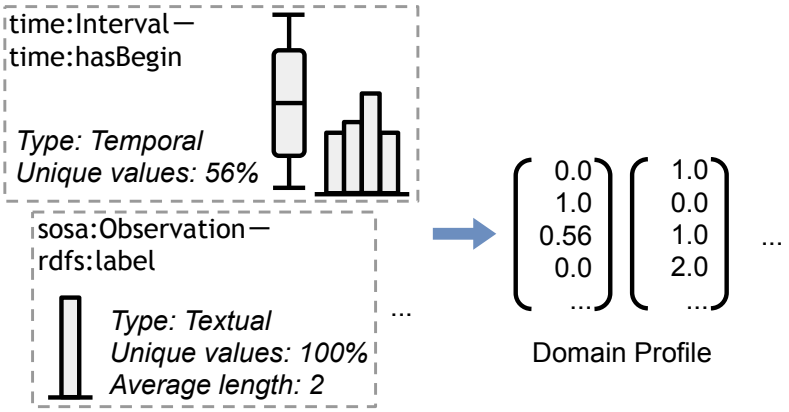


Figure 4.3. An example profile of the weather observation domain. The domain profile is represented as a set of feature vectors, each containing statistical features, such as value distributions. The domain profile can also be used for visualization of the data.

Consider the table in Fig. 4.1 that contains weather observation sensor data, separated by a tab character (↠). The table does not include column titles. As a human, we can observe that the first column refers to the air condition (*cloudy*, *clear*, *rain*). The second and third column may represent a time interval of the measurement (e.g., *16:30* and *17:00*). The fourth column containing the values *S2*, *S3*, and *S1* is hard to interpret without background knowledge.

To facilitate semantic table interpretation, *Tab2KG* relies on background knowledge regarding the target domain. Such background consists of two parts: a domain ontology and a domain profile.

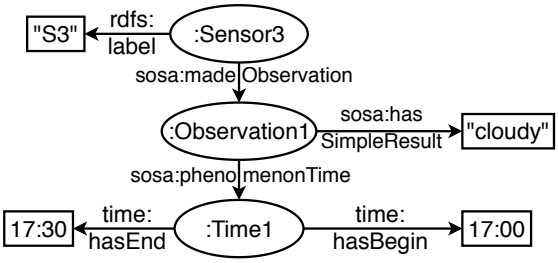


Figure 4.4. Correct mapping of the third line in Figure 4.1 to the ontology in Figure 4.2. For brevity, we omit `rdf:type` relations.

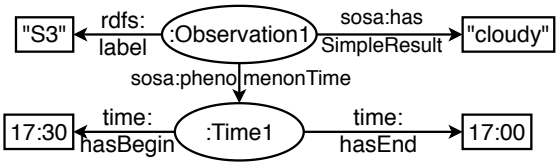


Figure 4.5. Incorrect mapping of the third line in Figure 4.1 to the ontology in Figure 4.2. For brevity, we omit `rdf:type` relations.

- The **domain ontology** models the specific domain of interest. In our running example, we use the Semantic Sensor Network Ontology[5] illustrated in Fig. 4.2. Amongst others, this ontology provides classes to model sensors, their observations, and corresponding time intervals.
- The **semantic domain profile** is a lightweight representation of typical value distributions for the ontology relations. In this example, such distributions can be obtained from prior weather observations. Fig. 4.3 gives an exemplary illustration of such a domain profile in the weather observation domain. Here, we illustrate statistical features of two observation properties (the beginning of the observation and the sensor label) using box plots and histograms. Such features can be represented as numerical feature vectors and included in a semantic domain description using DCAT and SEAS vocabularies.

Tab2KG enables the semantic table interpretation through profile matching, which maps table columns to the ontology relations. Given the mapping, *Tab2KG* transforms the table into the data graph shown in Fig. 4.4. As we can observe, the first three columns are mapped to the observations and their time intervals. The fourth column is mapped to the sensor labels.

The transformation process is challenging and potentially error-prone. For example, Fig. 4.5 illustrates a wrong transformation result, with an incorrect column mapping and an erroneous graph structure.

In this case, the sensor label "S3" was erroneously interpreted as an observation label. In addition, the beginning and end times are swapped. *Tab2KG* utilizes semantic profiles to avoid such interpretation errors.

[5] `sosa:` https://www.w3.org/TR/vocab-ssn/

4.1.3 Outline

The structure of this chapter is as follows: In Section 4.2, we discuss related work specifically relevant for *Tab2KG*. Then, in Section 4.3, we define the problem of semantic table interpretation, followed by the definition and creation of domain and data table profiles (Section 4.4). In Section 4.5, we describe our proposed *Tab2KG* approach and its implementation. We present evaluation setup and results in Section 4.6, followed by a discussion of our profile-based approach in Section 4.7. Finally, we provide a conclusion in Section 4.8.

4.2 Specific Background

In this section, we provide an overview of specifically relevant works in the areas of dataset profiling and semantic table interpretation.

Given the growth of data available on the Web and in industrial data lakes, there is a high demand for dataset profiling, e.g., for creating data catalogs [NUP16]. The profile features typically belong to several categories, including statistical observations at the instance and schema level [BEBB+18, AGN15]. Such features are not restricted to the initially defined schemas. For example, Neumaier et al. demonstrate how user interaction and search functionalities profit from the inclusion of spatio-temporal features into a dataset profile [NP19]. For tabular data, other approaches for dataset profile enrichment include the generation of table titles [HLY19] and schema labels [CJHD18]. The inferred relation-specific rules and observations can further verify the data quality and become part of dataset profiles [SLS+18, SRLJ19].

Recently, approaches to annotate tabular data with concepts from a knowledge base to predict column types gained increased attention. In the following, we introduce approaches for semantic table interpretation and the methods they use.

Instance-level lookup. Most semantic table interpretation tools require access to a target knowledge graph, as they link data table cells to its resources [RB17]. Such approaches on the instance-level have recently been driven by the SemTab Challenge [JRHE+20, JHE+20], which explicitly postulates a cell-entity annotation (CEA) task, where labels in data table cells are linked to entities in a target knowledge graph. The subsequent steps of column-type annotation (CTA) and columns-property annotation typically build upon the CEA results.

Several approaches are based on entity lookup (ColNet [CJRHS19], MantisTable [CDPRS20], LinkingPark [CKN+20], DAGOBAH [LT19, HLC+20], MTab [NKIT19], T2KMatch [RLB15], CSV2KG [SVDTO19], TableMiner+ [Zha17], and the work by Zhang et al. [ZMBR20]), with different (combined) query strategies, including URL matching [SVDTO19], (partial) string lookup [CDPRS20, HLC+20, CKN+20], string similarity [NKIT19, HLC+20, RLB15, ZMBR20], spelling correction [CKN+20] or the use of named entity linking tools [CDPRS20]. After linking data table cells to

resources in the knowledge graphs, the CTA is typically decided through voting or counting [NKIT19, CJRHS19, CKN+20], ranking [SVDTO19, CDPRS20, Zha17] or clustering [HLC+20]. ColNet and TableMiner+ apply learning strategies to reduce the number of lookup tasks required for detecting the class of the entities represented by a column. MantisTable and CSV2KG utilize concept graphs in their ranking to identify the most-specific sub classes. Also, identifying properties represented by the data tables typically relies on the CEA and additional knowledge graph lookups. For example, MTab does pairwise queries between entities identified in different columns to identify potential entity relations. To identify literal relations, MTab and TableMiner+ row-by-row compute data-type specific similarities between the literals in the target knowledge graph and the cell values. CSV2KG also involves the target ontology in this step. Sherlock [HHB+19] is a system that performs CTA and does not rely on CEA. However, it extracts column features for training a neural network, which is solely trained on DBpedia and explicitly predicts one of the selected DBpedia classes.

The reliance on entity linking with the target knowledge graph makes these approaches unsuitable in settings where data tables only contain previously unseen data, which is a common issue [RLOB16]. Even if the data instances in the data table are not entirely unknown, these approaches do not perform well when the number of matching entities drops [CJRHS19]. Another thing these approaches have in common is the reliance on a large underlying knowledge base such as DBpedia and stable lookup services. In contrast, *Tab2KG* does not require access to the target knowledge graph after the domain profile has been created.

Subject column detection. Several approaches [CDPRS20, Zha17] for semantic table interpretation assume the existence of a subject column, i.e., the main column of the data table where every other column is directly connected to. The subject column detection is typically identified through a set of statistic features. Approaches relying on a subject column do not consider the involvement of any classes which are not directly represented in the data table (for example, consider Fig. 4.1, but without the first column). *Tab2KG* utilizes a graph-based approach where such class relations can be found.

Column titles. Some data tables come with column titles, which may indicate respective classes or properties. Efthymiou et al. [EHRMC17] propose a method based on ontology matching, where one column title defines the class label, and other column titles represent property labels. Domain-independent Semantic Labeler [PAKS16], DAGOBAH [HLC+20] and TableMiner+ [Zha17] exploit column titles as one of their features. *Tab2KG* does not require any column titles. This way, we ensure the generalizability for data tables without headers and language-independence.

Data type restrictions. Data tables contain data of various types, and thus there are approaches specific to some of them. For example, EmbNum+ [NNIT19] transforms data table columns with numeric values into embedding vectors. Alobaid et al. demonstrate that using more fine-grained numeric data types increases semantic table interpretation performance for numeric column values [AKC19]. For the interpretation

of cross-lingual textual values, Luo et al. propose using several translation tools [LLCZ18]. *Tab2KG* aims at the interpretation of data tables as a whole without restricting to particular data types or languages and thus establishes profiles that do not depend on particular data types or languages.

User feedback. Instead of relying on fully-automated approaches for semantic table interpretation, which may be error-prone due to the challenges involved in this task, manual or semi-automated approaches rely on user feedback. Karma [KSA+12], Odalic [Kna17], and ASIA [CCDPP19] are interactive tools that let users decide on the correctness of suggested table annotations and thus achieve high precision, but demand both time and expertise from the user. *Tab2KG* is a fully-automated approach for semantic table interpretation that does not require user interventions.

Domain-independent semantic table interpretation. Domain-independent approaches are not restricted to specific target knowledge graphs. Instead, they learn domain-independent similarity features to generate the mapping. The SemanticTyper [RMKS15] scores similarity between columns and data type relations based on handcrafted features for numeric and textual values. Based on similar features, the Domain-independent Semantic Labeler [PAKS16] adopts machine learning and handcrafted features to predict the similarity between a column and a class in the domain knowledge graph. Taheriyan et al. [TKSA16a] generate a ranked list of potential column mappings learned from a sample of the domain ontology, which is then presented to the user. While both approaches are flexible concerning the target domain, *Tab2KG* aims to use only features present in the dataset profiles.

4.3 Problem Statement

In this section, we first formally define the concepts of a domain knowledge graph and a data table. Then, we present the task of semantic table interpretation.

The entities and relations in the domain of interest can be represented in a domain knowledge graph.

Definition 4.1. *A **domain knowledge graph** is a knowledge graph $G = (N, R)$ as defined in Definition 2.1, whose nodes N represent entities and literals, and whose edges R represent relations between these nodes in the specific domain.*

A domain knowledge graph consists of two sub graphs: a domain ontology and a data graph.

Definition 4.2. *A **domain ontology** $G_O = (N_O, R_O)$, $N_O \subset N, R_O \subset R$, where $N_O = C \cup D \cup P$ includes a set of classes C, a set of data types D, and a set of properties $P = P_d \cup P_o$, where P_d are data type properties, and P_o are object properties. Data type properties relate entities to literals. Object properties relate entities to each other.*

The relations represented by $R_O = R_{OC} \cup R_{OD}$ include class relations R_{OC} and data type relations R_{OD}:

- *R_{OC} is the set of class relations:*
  $R_{OC} \subseteq C \times P_o \times C$.
- *R_{OD} is the set of data type relations:*
  $R_{OD} \subseteq C \times P_d \times D$.

For example, in the excerpt of Semantic Sensor Network Ontology illustrated in Fig. 4.2, (`sosa:Sensor sosa:madeObservation sosa:Observation`) is a class relation and (`sosa:Sensor rdfs:label xsd:string`) is a data type relation.

Definition 4.3. *A **data graph** is a graph $G_D = (N_D, R_D)$, $N_D \subset N, R_D \subset R$. The nodes $N_D = C \cup D \cup E \cup L$ include classes C, data types D, entities E and literals L. Each literal $l \in L$ is assigned a data type $dt(l) \in D$. Within R_D, we distinguish between entity relations ($E \times P_o \times E$) and literal relations ($E \times P_d \times L$).*

A data table is defined as follows:

Definition 4.4. *A **data table** T is a $M \times N$ matrix consisting of M rows and N columns. A cell $T_{m,n}$, $m \in \{1, \ldots, M\}, n \in \{1, \ldots, N\}$, represents a data value. A row r_m is a tuple that represents a set of semantically related entities. A column c_n represent a specific characteristic of the entities in a row.*

For example, the data table illustrated in Fig. 4.1 contains $M = 7$ rows and $N = 4$ columns, where the columns represent the weather conditions, time intervals, and sensor labels, and each row contains three semantically related entities: an observation, a time interval, and a sensor. The column values can belong to different data types, including text, numeric, Boolean, temporal and spatial.

Semantic table interpretation is the task of transforming a data table into a data graph.

Definition 4.5. ***Semantic table interpretation**: Given a data table T and a domain knowledge graph G, create a data graph $G_D^T = (N_D^T, R_D^T)$ with nodes N_D^T and relations R_D^T. Its literal values $L^T \subseteq N_D^T$ are connected to the entities $E^T \subset N_D^T$ and represent the values in the literal columns of T. The entities E^T are connected via entity relations in R_D^T.*

4.4 Semantic Profiles

Semantic table interpretation in *Tab2KG* does not require any instance lookup in a domain knowledge graph. Instead, *Tab2KG* uses a sample domain knowledge graph

to create a lightweight semantic domain profile. This domain profile, together with a domain ontology, build reusable **domain background knowledge** that is later on used to interpret the data tables semantically. Note that the entities and literals in the sample domain knowledge graph do not need to overlap with the data tables' instances to be interpreted.

Tab2KG involves the creation of two types of profiles: *domain profiles* and *data table profiles*, both represented as feature vectors and described in a semantic data catalog to facilitate their reusability. Such profiles are inspired by the dataset profiles described in Section 2.3.5, where statistical features are defined as an important element of the dataset profiles taxonomy. In *Tab2KG*, the primary purpose of the domain and data table profiles is to enable effective and efficient access to the domain and table statistics for semantic table interpretation.

We present domain profiles in Section 4.4.1 and data table profiles in Section 4.4.2. We discuss profile features in Section 4.4.3. Then, in Section 4.4.4, we describe how we represent domain and data table profiles in a semantic, machine-readable way. Finally, in Section 4.4.5 we provide an example of a data catalog that includes semantic profiles.

4.4.1 Domain Profiles

For creating a domain profile, we make use of a domain knowledge graph G that contains representative values for the data type relations in the target domain. A **domain profile** is a set of data type relation profiles derived from G, where a data type relation profile is a set of statistical characteristics (features) of the literals covered by this data type relation in G's data graph G_D.

Definition 4.6. *The **data type relation profile** $\pi(r_D) \in \mathbb{R}^f$ of the data type relation $r_D \in R_{OD}$ is a vector that includes f features of the literal relations covered in the domain knowledge graph G.*

In brief, the profile of a data type relation r_D is a feature vector containing a set of statistics, computed using all literals corresponding to r_D.

To create a profile for the data type relation $r_D = (c, p_d, d)$, we utilize all literals $l \in G_D$, such that: $(e, p_d, l) \in R_D$, $(e, rdf : type, c) \in R_D$, and $dt(l) = d$.

In our running example, in Fig. 4.4, the data type relation (`sosa:Sensor rdfs:label xsd:string`) in the domain ontology corresponds to the literal relation (`:Sensor3 rdfs:label` "S3") in the data graph. Therefore, we use "S3" as one of the literals to create the data type relation profile.

4.4.2 Data Table Profiles

To facilitate semantic interpretation of a data table, we create a data table profile.

A **data table profile** is a set of column profiles, each representing a specific data table column. More formally, the profile of a data table T consists of a column profile $\pi(c_n), n \in \{1, \ldots, N\}$ for each table column $c_n \in T$ as defined in Definition 4.4.

A column profile is defined as follows:

Definition 4.7. *A **column profile** $\pi(c_n) \in \mathbb{R}^f$ of a column c_n is a vector of f statistical characteristics (features) of the values contained in that column.*

We create column profiles using literal values contained in the table columns.

Column profiles and data type relation profiles are created analogously and contain the same features, presented in Section 4.4.3.

4.4.3 Profile Features

Motivated by the RDF profile characteristics defined by Ellefi et al. [BEBB+18], we include data types, as well as completeness and statistical features described in the following into the profiles in *Tab2KG*. The selection is motivated by the expected feature effectiveness for semantic table interpretation, i.e., matching the domain and data table profiles. We demonstrate in our evaluation that these features can facilitate an effective matching in several application domains. This feature set can be further extended to include relevant characteristics in specific domains.

- **Data type:** We represent data types as binary profile features. We include fine-granular data types to facilitate the precise matching of domain and data table profiles. The following data type taxonomy includes the most common cases observed in our evaluation domains.
  - **Text:** Categorical, URL, Email, Other
  - **Numeric:** Integer, Decimal / Sequential, Categorical, Other[6]
  - **Boolean**
  - **Temporal:** Date, Time, Date Time
  - **Spatial:** Point, Linestring, Polygon

  A data type relation or column can be assigned multiple (fine-granular) data types (e.g., integer and categorical). We provide technical details regarding the identification of fine-granular data types later in Section 4.5.8.
- **Completeness:** We include the number of non-null values as a completeness indicator.

[6] following the taxonomy defined in [AKC19]

- **Basic statistics:** We include the number of values, the number of distinct values, as well as the average length and the average number of digits in the literals.
- **Histograms:** Histograms are an effective means for RDF data summarization [HHK+10]. We create a histogram for a given number of buckets as part of the data type relation profile or column profile. As features, we add the number of literals in each bucket, in the increasing order of bucket ranges. For histogram creation, we remove the outliers detected using the interquartile range (1.5 IQR) rule.
- **Quantiles:** We add quartiles and deciles to the profile (including minima and maxima). In addition, we add the number of outliers detected using the 1.5 IQR rule.

To derive numerical features, we transform literals into numbers. The features of textual data type relations are computed based on the textual values' lengths. Temporal values are transformed into timestamps. For spatial values, we consider the line string length or the polygon area, respectively.

4.4.4 Semantic Profile Representation

Domain and data table profiles can be represented as **semantic profiles** in RDF, as an extension of the Data Catalog Vocabulary (DCAT)[7] and the SEAS Statistics ontology[8].

Within the DCAT vocabulary, a data catalog (`dcat:DataCatalog`) consists of datasets (`dcat:Dataset`), where a dataset is a collection of data, published or curated by a single agent. In the context of *Tab2KG*, both the domain knowledge graph and the data tables can be represented using `dcat:Dataset`. We extend the descriptions of datasets in a *Tab2KG* data catalog to include semantic profiles. For example, we introduce an `Attribute` class representing the data table columns and data type relations. We make the definitions of this vocabulary available online[9].

Fig. 4.6 provides an overview of the classes involved in representing semantic profiles. A `dcat:Dataset` in a *Tab2KG* data catalog includes several attributes. In the case of a data table profile, these attributes the columns. In the case of a domain profile, these attributes are the data type relations. These attributes are assigned the profile features, as presented in Section 4.4.3:

1. Data types: Data type assignments follow the previously mentioned taxonomy.

[7] https://www.w3.org/TR/vocab-dcat-2/
[8] https://ci.mines-stetienne.fr/seas/StatisticsOntology
[9] https://github.com/sgottsch/Tab2KG

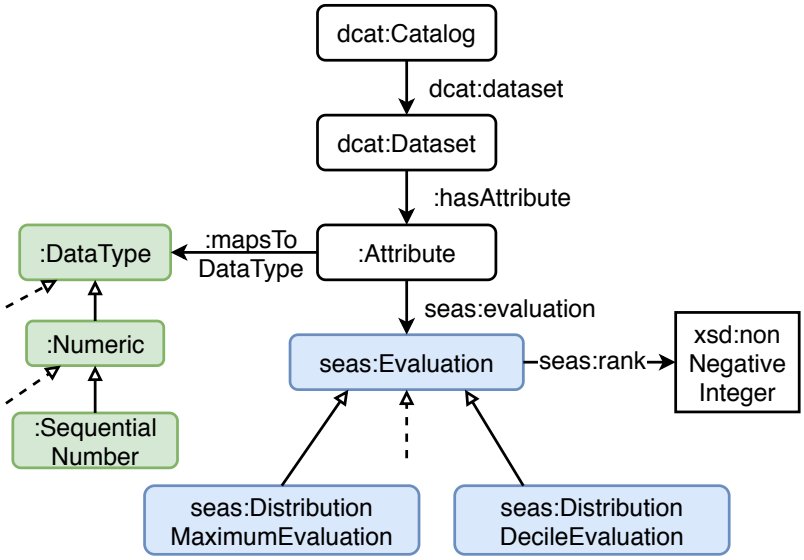


Figure 4.6. Classes and properties used for describing a semantic profile. ⇾ marks `owl:subClassOf` relations. Dashed arrows indicate the existence of further classes which are not included in this excerpt. The two feature types (data types and numeric features) are grouped by different colors.

2. Numeric features: The numeric profile features are represented through subclasses of `seas:Evaluation`. In the case of quartiles, deciles, and histograms, the values come with a rank. For example, we can denote the second decile using `seas:rank` 2.

In the case of domain profiles, the existing mapping to the domain ontology can be modeled by connecting attributes to their corresponding classes, and data type properties [GTK+19].

Note that such semantic profiles do not only enable semantic table interpretation but can also be used to provide lightweight dataset visualizations, e.g., through box plots (quartiles) or bar charts (histograms).

4.4.5 Running Example: Weather Data Catalog

An excerpt of an example data catalog for our running example from the weather observation domain introduced in Section 5.1.1 is shown in Fig. 4.7. The data catalog identified as `WeatherCatalog` includes two data tables (`RainData` and `AirData`). Here, the `AirData` data table has two columns, one of them with a column profile feature denoting the maximum value of the observation end time.

With *Tab2KG*, we can directly utilize this catalog for semantic table interpretation. Both example data tables can be interpreted through their data table profiles when a domain profile of the weather observation domain is provided.

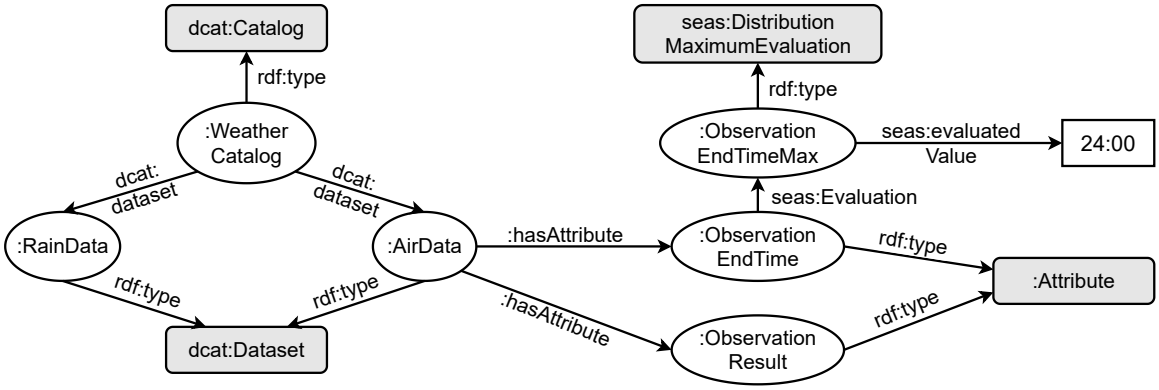


Figure 4.7. Running example: Excerpt of a weather data catalog containing two data tables and an exemplified column profile feature denoting the maximum end time value (`ObservationEndTimeMax`).

4.5 *Tab2KG*: Approach

To facilitate semantic table interpretation, *Tab2KG* utilises a schema graph that describes the specific domain (e.g., weather observations) and domain background knowledge encoded in a domain profile. In this section, we describe the interpretation process and how such profiles are created and matched.

4.5.1 Approach Overview

Fig. 4.8 provides an overview of our proposed *Tab2KG* approach to semantic table interpretation, where a data graph G_D^T is created from a data table T. To facilitate the interpretation, *Tab2KG* utilizes domain background knowledge that includes a domain ontology G_O and a domain profile. The domain profile is generated in a pre-processing step from a domain knowledge graph G.

In brief, the *Tab2KG* pipeline consists of the following steps:

1. **Domain Profile Creation**: In a pre-processing step, we create a domain profile from a domain knowledge graph G.
2. **Data Table Profile Creation**: We create a profile of the input data table T.
3. **Column Mapping**: We generate candidate mappings between the columns of T and the data type relations in G_O using the domain profile, the data table profile, and a one-shot learning mapping function.
4. **Data Graph Creation**: We use the candidate column mappings and the domain ontology G_O to create a data graph G_D^T representing T's content.

In the following, we describe these steps in more detail.

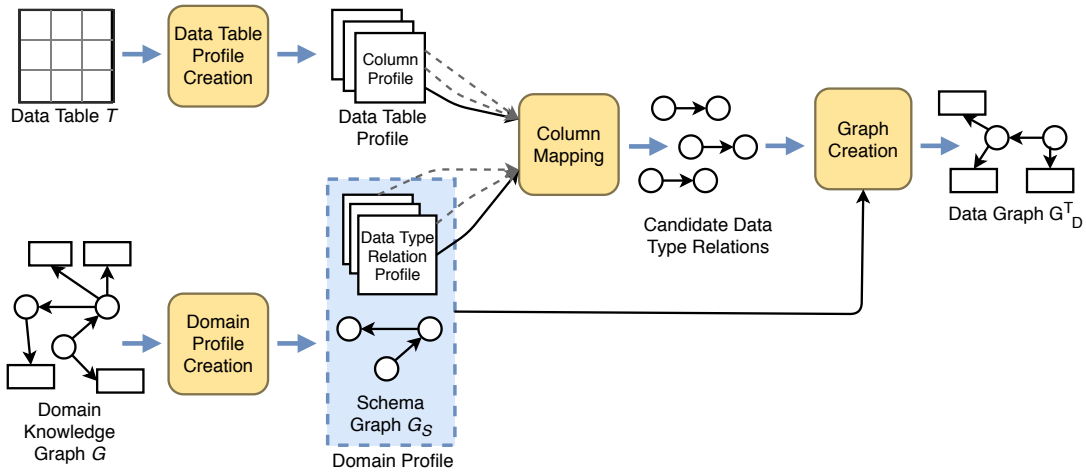


Figure 4.8. An overview of semantic table interpretation with *Tab2KG*. Input is a data table T and a domain knowledge graph G. The output is a data graph G_D^T that represents the content of T as a data graph.

4.5.2 Domain Profile Creation

The semantic table interpretation in *Tab2KG* requires the availability of a domain profile. This profile can be inferred from a domain knowledge graph G as described in Section 4.4.1. The domain profile is created by computing the feature values given the literal relations in the domain knowledge graph. This profile can be used as a lightweight domain representation. The domain profile can be created in a pre-processing step and become available as part of a data catalog as described in Section 4.4.4.

Note that the domain profile does not contain any entities or literals from the domain knowledge graph G. The domain knowledge graph is not directly used for the semantic table interpretation.

4.5.3 Data Table Profile Creation

From the input data table T, we create a data table profile by computing the profile features based on the column values as described in Section 4.4.2.

4.5.4 Column Mapping

With the help of the domain profile and the data table profile, we create *column mappings*.

Definition 4.8. *A **column mapping** is a mapping from a column c_n in a data table T to a data type relation $r_D \in R_D$ in the domain ontology G_O: $c_n \mapsto r_D$.*

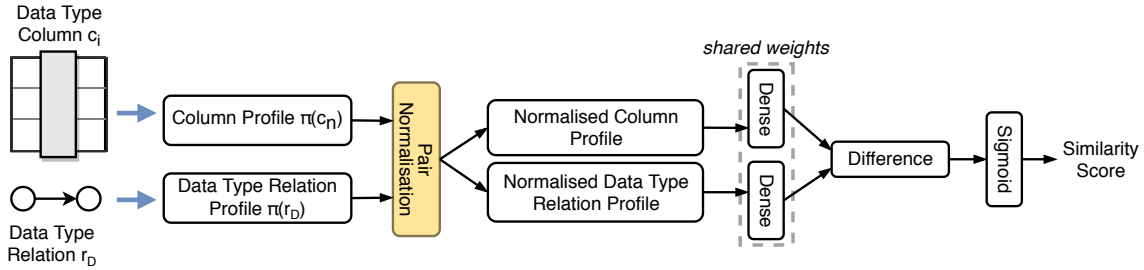


Figure 4.9. Architecture of the mapping function to predict the similarity between a column c_n and a data type relation r_D.

For example, we can create a mapping from column c_2 of the data table illustrated in Fig. 4.1 to the data type relation (`time:Interval` - `time:hasBegin` - `xsd:time`) in the ontology illustrated in Fig. 4.2.

Within the *Tab2KG* pipeline shown in Fig. 4.8, we use a mapping function that creates a set of candidate column mappings M_{c_n} for each column c_n in the data table T. Given a column profile and a data type relation profile, the mapping function returns a similarity score in the range $[0, 1]$. The mapping is created by detecting all data type relation profiles $\pi(r_D)$ similar to the respective column profile $\pi(c_n)$. In this step, we only consider mappings between columns and data type relations of the same data type (numeric, textual, temporal, spatial or Boolean).

The architecture of the column mapping function is shown in Fig. 4.9. First, it takes two profiles as an input and performs a joint normalization, i.e., the features are normalized in a range between 0 and 1 concerning the sum of the values in both profiles. Then, we follow the idea used for one-shot learning for image classification [KZS15]. Here, the task is not only to classify images into known classes (e.g., many images showing tigers) but also to generalize towards new classes (e.g., new images showing lions). That means, the underlying classifier needs to acquire features which enable the model to successfully generalize. This is done by inducing a metric that represents the domain-independent similarity between two input feature vectors (e.g., between an unknown image and a single image showing a lion).

As we cannot train a classifier on known classes (in contrast to domain-specific approaches such as Sherlock [HHB+19] and ColNet [CJRHS19]), we are in a one-shot learning setting as well: We may learn how to map column profiles to known data type relations. But when facing a new domain, our classifier needs to generalize towards unseen data type relations. In *Tab2KG*, the similarity between a column and a data type relation is predicted based on the experience of the similarity of other profiles learned earlier. The score to measure such similarity is facilitated by a Siamese network that encodes both profiles using the same weights and then predicts the similarity score based on the difference between the two profile encodings. As in [KZS15], we use Rectified Linear Units for the hidden layers and a Sigmoid output layer.

4.5.5 Data Graph Creation

Given a set of candidate column mappings M_{c_n} with similarity scores for each column c_n in the input data table T, we now assign each table column a data type relation in a greedy manner. First, we take the column mapping with the highest similarity score. Then, we remove all candidate column mappings with the particular column or data type relation. These two steps are repeated until all columns are assigned a data type relation.

From the chosen column mappings set, we create the data graph G_D^T that contains all data type relations resulting from the chosen mapping. G_D^T needs to adhere to the following four conditions:

1. The data graph covers all literal columns of T, and each literal column has exactly one mapping to a data type relation.
2. The set of entities in a table row is connected via entity relations.
3. G_D^T is minimal, i.e., no relation can be removed without invalidating the previous two conditions.
4. Each class relation represented by G_D^T is connected to at least one class that is part of M_{c_n}. This condition ensures semantic closeness of the data table columns and reduces the number of potential paths in the graph.

4.5.6 Creation of Training Instances for Column Mapping

For the computation of the column mapping function, we utilize a Siamese network trained once in a pre-processing step. This training process requires the extraction of positive and negative training instances. Following Definition 4.5 (see also Fig. 4.8), this step requires a set of (G, T, G_D^T) triples, where G is the domain knowledge graph, T is the data table and G_D^T is the resulting data graph. For each triple (G, T, G_D^T), each pair of data type relations in G and a column in T is a positive training instance. We select the remaining (data type relation, column) pairs from the same knowledge graph G as negative instances.

In the first step, we synthetically create a set of (G, T, G_D^T) triples for the model training, intending to have a large dataset of positive and negative examples derived from existing data tables and knowledge graphs. Such data is difficult to obtain, except for manually created, task-specific research datasets [PAKS16], which are not large enough for training deep neural networks and do not provide enough topical and structural diversity. Therefore, we utilize existing knowledge graphs to create training data.

Given a set of knowledge graphs, we create a new dataset of triples (G_1, T, G_2^T). Each input knowledge graph G is disjunctly split into two KGs: G_1 and G_2. G_1

represents the domain knowledge graph, while G_2 is transformed into a data table T. The transformation of G_2 into T is based on a set of domain ontology templates. A template is a directed tree with up to k nodes, where k is a parameter. The nodes and edges of these trees are placeholders for classes and properties. A set of trees is transformed into domain ontologies by replacing these placeholders with the classes and properties of G_2. From the knowledge graph G_2, a data graph G_2^T is extracted and transformed into a data table T to create the triple (G_1, T, G_2^T).

We aim to retrieve a set of heterogeneous data tables that represent the original knowledge graph characteristics. Therefore, the data table creation process incorporates several stochastic decisions in proportion to the knowledge graph statistics:

- Entities and entity relations (and consequently, the literal relations) in G are split at a random ratio between 25% and 75% into G_1 and G_2, whereas their domain ontologies remain the same.
- During the template creation, classes, and properties are assigned randomly to a domain ontology template, proportionally to their occurrence rate in G.
- Data type relations are added in the same manner, under the condition that each leaf node has to be connected with at least one data type relation. After adding the minimal required number of data type relations, we add data type relations as long as any of them are left and if a randomly generated number between 0 and 1 exceeds a predefined threshold δ.

4.5.7 Running Example: Data Table Creation

For our running example introduced in Section 4.1.2, Fig. 4.10 illustrates the transformation of a domain knowledge graph G and a domain ontology τ into a data table T with two rows and two columns. In this specific minimal example, the data graph G_D^T is identical with the input knowledge graph G, as τ equals the domain ontology of G_2.

4.5.8 Implementation

Tab2KG is implemented in Java 1.8. The Siamese network is trained and applied using Keras in Python 3.7. We load knowledge graphs using Apache Jena[10]. We represent the column mappings inferred by *Tab2KG* in the RDF Mapping Language (RML) [DVSC+14]. RML definitions are then utilized to materialize the data graph. Data tables are provided as CSV files; knowledge graphs and data graphs as Turtle (*.ttl*) files.

[10] https://jena.apache.org/

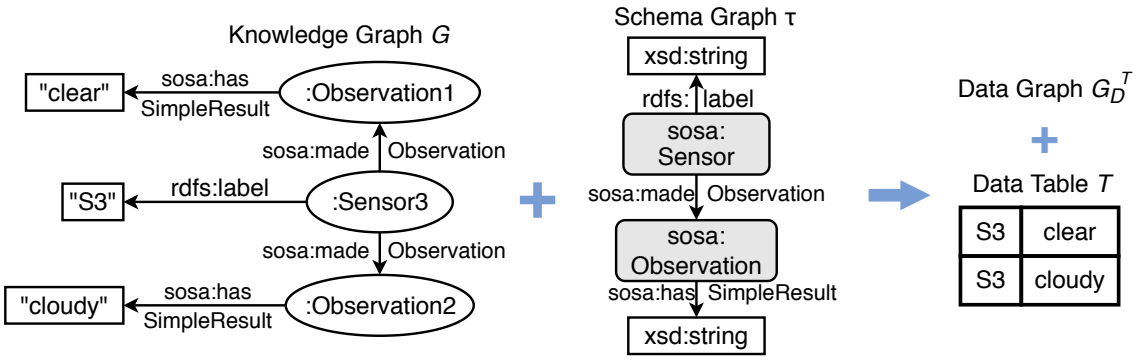


Figure 4.10. Creation of a data table T and a data graph G_D^T from a domain knowledge graph G and a domain ontology template τ. For brevity, we omit type relations. In this particular example, G_T^D equals G.

4.5.9 Data Table Pre-Processing

Each data table interpreted in *Tab2KG* runs through three pre-processing steps:

1. **Data type identification**: For each table column, we identify its data type(s) by trying to parse more than 90% of the values as numeric, Boolean, spatial (Well-Known Text or Well-Known Binary format [ISO16]) or temporal. If that is not possible, we assign the column to the text data type. For the more fine-grained data types mentioned in Section 4.4, we utilize regular expressions (text: URL or email), follow the algorithms proposed by Alobaid et al. [AKC19] (numeric: sequential or categorical) or analyze the parsed objects (temporal: date, time, date-time; spatial: point, line string, polygon). We follow the algorithm in [AKC19], using the threshold of not more than 20 different categories to detect categorical text values.

2. **Key column detection**: When we transform a data table into a data graph based on the RML mapping, we create new entities. In RDF, each entity is identified by a Unique Resource Identifier (URI). It is important to understand how to create these URIs, as we need to re-use URIs referring to the same entity: For example, each row in our running example in Fig. 4.1 forms a new instance of `sosa:Observation`, together with a new URI (e.g., `sosa:Observation7`), but there should only be three different `sosa:Sensor` URIs: `sosa:SensorS1`, `sosa:SensorS2`, `sosa:SensorS3`, created using the literal values assigned to the data type property `rdfs:label`.

   To create URIs, we detect data type relations representing the unique literal values of an entity as follows: (i) the data type relation is used on all instances of the class exactly once, and the literal values are unique across their instances. Currently, we do not consider the combinations of literal relations as identifiers [HQRA+13]; we leave such combinations for future work.

3. **Identifier generation**: RML transformation requires referenceable columns and instances in data tables. Therefore, we automatically generate identifiers for each column and row of the data table. If available, column names are used as part of the column identifier.

Mapping Representation in RML

We utilize RML for storing column mappings inferred by *Tab2KG* in a machine-readable format such as Turtle. The RML defines subject maps that specify how to generate subjects for each row of the data table and predicate-object maps to create predicate-object pairs. In *Tab2KG*, the inferred column mappings are translated into the RML definitions according to the following four steps:

1. We create one instance of `rml:source` and `csvw:Table` each, denoting relevant characteristics for parsing the data table T (file location, delimiter, ...).
2. For each class part of the data graph G_D^T, we create a new instance of `rr:TriplesMap`, together with a `rr:subjectMap` that denotes the class as well as the target node URIs.
3. For each column mapping $c_n \mapsto r_D$, we create a `rr:predicateObjectMap` denoting the source column c_n, the data type property and a reference to the data type relation r_D.
4. For each class relation $r \in R_{OC}$ in the domain ontology G_O, we create a `rr:predicateObjectMap` connecting the respective entities and the object property.

Running Example: RML Mapping Definitions

Listing 4.1 provides an example of the RML definitions that were automatically generated for our running example introduced in Section 4.1.2 – without the time intervals, for brevity. Instances of `sosa:Sensor` and `sosa:Observation` are created alongside their relations. The sensor labels in the third column were detected as identifiers, i.e., we create node URIs as *https://www.w3.org/TR/vocab-ssn/Sensor*{*col3*}[11].

Listing 4.2 provides the resulting Turtle file representing the knowledge graph inferred from the input data table. The correct mapping of the third line shown in Fig. 4.4 is contained here.

[11]The RML template definitions do not allow to use prefixes.

```
@prefix rr: <http://www.w3.org/ns/r2rml#>.
@prefix rml: <http://semweb.mmlab.be/ns/rml#>.
@prefix ex: <http://example.com/resource/>.
@prefix csvw: <http://www.w3.org/ns/csvw#>.

ex:File a rml:source ;
  rml:source ex:FileSource ;
  rml:referenceFormulation <http://semweb.mmlab.be/ns/ql#CSV> .

ex:FileSource a csvw:Table;
  csvw:url "sky_sensors.tsv" ;
  csvw:dialect [
    a csvw:Dialect;
    csvw:delimiter "↠";
  ] .

ex:Mapping0 a rr:TriplesMap ;
  rml:logicalSource ex:File ;
  rr:subjectMap [
    rr:class sosa:Sensor ;
    rr:template "https://www.w3.org/TR/vocab-ssn/Sensor{col3}" ;
  ];

  rr:predicateObjectMap [
    rr:predicate rdfs:label ;
    rr:objectMap [
      rml:reference "col3";
    ]
  ] ;

  rr:predicateObjectMap [
    rr:predicate sosa:madeObservation ;
    rr:objectMap [
     rr:template "https://www.w3.org/TR/vocab-ssn/Observation{rowNumber}";
    ]
  ] .

ex:Mapping1 a rr:TriplesMap ;
  rml:logicalSource ex:File ;
  rr:subjectMap [
    rr:class sosa:Observation ;
    rr:template "https://www.w3.org/TR/vocab-ssn/Observation{rowNumber}" ;
  ] .
```

Listing 4.1: A working example of an RML file transforming the data table given in Fig. 4.1 ("sky_sensors.tsv") into the data graph indicated in Fig. 4.4. To perform the transformation, the table needs to be pre-processed: the column titles "col0",...,"col3" were added, and a "rowNumber" column. For brevity, we skip the mapping of the second and third column to `time:Interval`.

```
<https://www.w3.org/TR/vocab-ssn/Observation0>
  a <http://www.w3.org/ns/sosa/Observation> .

<https://www.w3.org/TR/vocab-ssn/Observation1>
  a <http://www.w3.org/ns/sosa/Observation> .

<https://www.w3.org/TR/vocab-ssn/Observation2>
  a <http://www.w3.org/ns/sosa/Observation> .

<https://www.w3.org/TR/vocab-ssn/SensorS2>
  a <http://www.w3.org/ns/sosa/Sensor>;
  <http://www.w3.org/2000/01/rdf-schema#label> "S2";
  <http://www.w3.org/ns/sosa/madeObservation>
    <https://www.w3.org/TR/vocab-ssn/Observation0> .

<https://www.w3.org/TR/vocab-ssn/SensorS3>
  a <http://www.w3.org/ns/sosa/Sensor>;
  <http://www.w3.org/2000/01/rdf-schema#label> "S3";
  <http://www.w3.org/ns/sosa/madeObservation>
      <https://www.w3.org/TR/vocab-ssn/Observation1>,
      <https://www.w3.org/TR/vocab-ssn/Observation2> .
```

Listing 4.2: The Turtle file resulting from running the RML file in Listing 4.1 to transform the table given in Fig. 4.1 into a data graph. For brevity, we only consider the first three lines of the data table.

4.6 Evaluation

The goal of the evaluation is to assess the performance of *Tab2KG* concerning the semantic table interpretation effectiveness. In this section, we describe the datasets utilised for the evaluation and provide the evaluation results.

4.6.1 Datasets

For training and evaluating *Tab2KG*, we use several datasets.

The Siamese network training requires a dataset that spans over multiple domains and domain knowledge graphs, respectively, to ensure generalization. As the datasets typically used for semantic table interpretation only target a single domain or a cross-domain knowledge graph such as DBpedia, we created a new synthetic dataset automatically extracted from GitHub repositories dealing with knowledge graphs.

For testing, we consider the GitHub dataset and well-established datasets for semantic table interpretation that target specific domains (soccer and weapon advertisements), and DBpedia as a cross-domain knowledge graph.

Synthetic GitHub Dataset

To gain a dataset that covers a large variety of domains and schemas, we collect knowledge graphs from GitHub. The GitHub advanced code search[12] provides access to millions of data repositories. We selected files larger than 5KB with the specific file extensions[13] that contain the text "`xsd`" or "`XSDSchema`". To ensure the heterogeneity of our dataset, we limited the number of files per GitHub repository to three. Each file that was successfully parsed as a knowledge graph with more than 50 statements including at least 25 literal relations was added to our dataset. This way, we obtained 3,922 files.

We set the parameter for maximum tree size $k = 3$, and the parameter for adding data type relations $\delta = 0.2$. The knowledge graphs set was split into a training set (90%) and a test set (10%). Knowledge graphs from the same repository were not included in the same set.

Test Datasets

We use the following datasets for evaluating our approach:

- **GitHub (GH):** The test split of the synthetic GitHub dataset, without a restriction on the used vocabularies.
- **Soccer (So):** 12 data sources regarding soccer players and their teams, annotated with the *schema.org* vocabulary [PAKS16].
- **Weapon Ads (WA):** 15 data sources about weapon advertisements, annotated with the *schema.org* vocabulary [TKSA16b].
- **SemTab (ST):** A collection of data tables extracted from the T2Dv2 web table corpus, Wikipedia and others, annotated with the DBpedia ontology [JRHE+20].
- **SemTab Easy (SE):** A subset of ST, including only data tables whose columns are mapped to one class only. Only classes appearing in the *T2KMatch* corpus [RLB15] are included.

For all data tables contained in these datasets, we set the following constraints:

1. The input table file is parseable as a CSV file without errors.
2. There are no classes that are instantiated multiple times in the same row. This condition is to avoid cyclic structures. We discuss limitations regarding cyclic structures in Section 4.7.

[12] https://github.com/search/advanced
[13] `ttl`, `rdf`, `nt`, `nq`, `trix`, `n3`, `owl`

Table 4.1. Datasets used in the evaluation. # is the number of (G, T, G_D^T) triples. The other columns contain average values. For example, the tables in ST dataset have about four columns on average.

		Tables	Domain Knowledge Graphs	
Dataset	**#**	**Columns**	**Data Type Relations**	**Class Relations**
GitHub (Training)	867	2.18	3.03	1.35
GH: GitHub (Test)	98	2.24	3.16	1.44
So: Soccer	15	6.25	9.38	2.75
WA: Weapon-Ads	16	10.57	11.2	4.4
ST: SemTab	233	4.09	5.54	1.26
SE: SemTab Easy	125	4.2	5.53	0.0

3. There is no pair of columns with identical values. This condition is to avoid randomness during the evaluation.

It is important to emphasize the difference in the evaluation setting compared to typical evaluation using the previously mentioned datasets such as ST: In the experiments conducted by [PAKS16, CKN+20, Zha17], target general-purpose knowledge graphs such as DBpedia or Wikidata are given. Each data table in the test set is then mapped to the nodes in such a knowledge graph. In our evaluation setting, we assume that no data instances are given, i.e., an instance lookup is not possible. Instead, a domain profile and a domain ontology are provided. For evaluation, we select data table pairs, such that one data table mimics the domain knowledge graph from which we can derive a domain profile.

Following our setting defined in Definition 4.5 and illustrated in Fig. 4.8, the datasets are transformed into a set of triples (G, T, G_D^T), consisting of a data table T, a domain knowledge graph G and the mapping definition which transforms data table T into the data graph G_D^T. Technically, we extract a set of test instances, consisting of (i) a *.ttl* file representing the domain knowledge graph G, and (ii) a *.csv* file representing the data table T and a *.rml* file representing the mapping from T to G's domain ontology. To transform the datasets into such test instances, we identify pairs of data tables where the columns of the first data table are a subset of the second data table's columns. Then, the second data table represents the domain knowledge. Table 4.1 provides an overview of the datasets used during training and evaluation under these conditions.

We make the scripts to extract such evaluation datasets and the code for training the Siamese network publicly available[14].

[14] https://github.com/sgottsch/Tab2KG

4.6.2 Baselines

We compare *Tab2KG* against the following three semantic table interpretation baselines:

- ***DSL***: The Domain-independent Semantic Labeler [PAKS16] uses logistic regression on a set of hand-crafted features; some of them compare value pairs at the instance-level. It has been shown to outperform previous approaches such as the SemanticTyper [RMKS15]. We train *DSL* on the same GitHub training data set as *Tab2KG*.
- ***DSL****: The Domain-independent Semantic Labeler without using value similarity at the instance level. In contrast to *Tab2KG* and the other baselines, *DSL* utilizes a domain-specific data graph. As *Tab2KG* is solely based on the domain profile, *DSL* is in an advantageous setting that does not entirely reflect our setting. Therefore, we remove features at the instance-level for the *DSL** baseline.
- ***T2KMatch***: *T2KMatch* [RLB15] performs semantic table interpretation on the instance level. In contrast to other approaches [CDPRS20, NKIT19, Zha17] that rely on a costly knowledge graph lookup at runtime, *T2KMatch* creates an index over the instances of frequently used DBpedia classes and is thus commonly used as a baseline for semantic table interpretation approaches [ZMBR20, EHRMC17, CJRHS19]. It combines a ranking of entities found in the lookup phase for column type identification and data type-specific similarity measures (Levenshtein distance for strings, deviation for numbers, and deviation of years for dates) for property identification. *T2KMatch* assumes that a table only describes one entity class at a time and thus does not consider class relations.

Now, we describe the training performance and the evaluation results based on the evaluation setup described before.

4.6.3 Accuracy of the Column Mapping Function

We train our Siamese network for $1,000$ epochs with a batch rate of 100 and a learning rate of 0.00006, following Hsiao et al.'s approach for one-shot image classification [HKLT19]. We use 256 dimensions for the hidden layer. 10% of the training dataset are used for validation. The feature vectors contain histograms with 10 buckets.

After training for all epochs on the synthetic training dataset, the Siamese network has an accuracy of 0.84 and a loss (binary cross-entropy) of 0.48 on the validation set. On the test set, it achieves an accuracy of 0.76, when treating scores of greater than 0.5 as candidates for column mapping.

Table 4.2. Semantic table interpretation performance of *Tab2KG*, compared to the baselines on five datasets. We report the accuracy, i.e. the percentage of correctly identified data type relations (R_{OD}) and class relations (R_{OC}) in the datasets.

	GH	So	WA	ST	SE	Average
DSL	0.89	0.65	0.38	0.62	0.61	0.67
DSL*	0.87	0.43	0.44	0.66	0.71	0.70
T2KMatch	-	-	-	-	-	0.44
Tab2KG	0.88	0.64	0.48	0.78	0.78	**0.79**

4.6.4 Semantic Table Interpretation Results

We evaluate the performance of the semantic table interpretation achieved by *Tab2KG* compared to the baselines. Table 4.2 shows how the approaches perform on the different datasets, measured using accuracy, i.e., the percentage of the columns correctly mapped to data type relations and correctly identified class relations. We do not evaluate the performance of *T2KMatch* on other datasets than SE, as *T2KMatch* assumes one entity class per table only.

As we can observe in Table 4.2, the accuracy of the approaches varies considerably across the datasets, which can be explained by the different dataset characteristics shown in Table 4.1. In all cases except for the GH and So datasets, where *Tab2KG* and *DSL* show similar performance, *Tab2KG* achieves higher accuracy than the baselines concerning column mapping. Even though *Tab2KG* utilizes less information than *DSL*, *Tab2KG* performs better by 12 percentage points on average on this task.

Surprisingly, *DSL* is also outperformed by *DSL** on three of the five datasets (WA, ST, SE). To explain this behavior, we have computed the percentage of table values that also appear in the mapped data table relations: GH (33.38%), So (16.92%), WA (2.65%), ST (14.67%), SE (10.63%). This observation shows that profile-based semantic table interpretation can outperform instance-level approaches when the overlap between the data table and the instances in the domain knowledge graph is low.

Second, we assess the results of the column mapping (percentage of correctly mapped columns to the data type relations R_{OD}) and the graph creation (correctly identified class relations R_{OC}) in isolation. We report the results achieved by *Tab2KG* in comparison to the baselines in Table 4.3. In the case of column mapping, *Tab2KG* performs best on average, outperforming *DSL* by 10 percentage points. In general, the class relation mapping results in less accuracy than the column mapping. One reason is that errors propagate along the pipeline, i.e., a wrongly mapped data type relation invokes an erroneous class relation mapping.

Table 4.3. Semantic table interpretation performance of *Tab2KG* in detail, compared to the baselines on five datasets, reported as the accuracy of class relations (R_{OC}) and data type relations (R_{OD}). Averages are computed in relation to the number of class relations and data type relations in the datasets, respectively.

	GH		**So**		**WA**		**ST**		**SE**	**Avg**	
	R_{OC}	R_{OD}	R_{OC}	R_{OD}	R_{OC}	R_{OD}	R_{OC}	R_{OD}	R_{OC}	R_{OC}	R_{OD}
DSL	0.93	0.62	0.42	0.46	0.45	0.40	0.65	0.70	0.71	0.67	0.57
DSL*	0.94	0.71	0.60	0.81	0.33	0.47	0.60	0.87	0.61	0.62	**0.71**
T2KMatch	-	-	-	-	-	-	-	-	0.44	0.44	-
Tab2KG	0.92	0.71	0.61	0.73	0.59	0.24	0.77	0.89	0.78	**0.77**	0.64

4.6.5 Error Analysis

By inspection of the results, we have identified two typical sources of erroneous results in *Tab2KG*: (i) Value formatting: For example, the soccer dataset has data tables with column values such as "Germany", whereas the domain knowledge graphs had "GER" as a country label. Thus, the respective profile features were highly different. In the case of high-quality domain knowledge graphs that, for example, distinguish between labels and abbreviations, the error rate should be lower. (ii) Data type differences: For example, the SemTab dataset has elevations of mountains both denoted via integer values ("5291") and as text ("2,858 ft (871 m)"). Again, the proper use of an ontology and its `rdfs:range` constraints on properties should alleviate this problem.

Overall, our evaluation results demonstrate that domain profiles in combination with zero-shot learning adopted by *Tab2KG* are an effective method for semantic table interpretation. This method does not require any instance lookup and achieves the highest accuracy on several datasets compared to the baselines.

4.7 Limitations

We identified few limitations of *Tab2KG*, which can be attributed to the idea of using semantic lightweight dataset profiles, without requiring knowledge about particular data instances.

4.7.1 Column Mapping without Knowing the Dataset Instances

In contrast to approaches that perform semantic table interpretation at the instance level, i.e., with the help of the instance lookup in the domain knowledge graph, *Tab2KG* derives column mappings from statistical features in the domain profile. We have identified two limitations to this approach:

- Cyclic class relations: Currently, we do not address cyclic relations in the domain ontology, as for example (`dbo`[15]`:Event dbo:nextEvent dbo:Event`). Consider Fig. 4.11, where the second column provides the follow-up event of the event in the first column. Even if *Tab2KG* identifies the correct mapping for the third column to the property `dbo:locationCity`, we cannot tell if the third column maps to the location of the entity in the first or the second column.
- Class relations connecting the same classes: We do not have a decision criterion for distinguishing between class relations that connect the same subject and object classes. For example, consider the two object properties `dbo:leader` and `dbo:viceLeader` mapped to the second column in Fig. 4.12, both connecting countries to politicians. Only via statistical features extracted from the data table (which may only include the politician's name) it does not appear possible to decide if Kamala Harris is the president or the vice president of the United States.

```
Olympics 2004 ↠ Olympics 2008 ↠ Athens
Olympics 2012 ↠ Olympics 2016 ↠ London
```

Figure 4.11. Cyclic class relation: Did the London Olympic Games happen in 2012 or in 2016?

```
USA    ↠ Kamala Harris
Russia ↠ Michail Mischustin
```

Figure 4.12. Class relations connecting the same classes: Is Kamala Harris the president or the vice president of the US?

4.7.2 Correlations between Columns and Data Type Relations

Our data table profiles consist of column profiles, i.e., the features of the single columns are computed in isolation (the same applies to domain profiles and data type relations). Such column profiles can be efficiently computed and added to the dataset profile. However, the dependencies between columns may hold additional knowledge. Consider the running example in Fig. 4.1 where the time in the second column (begin time) does always precede the time in the next column (end time), i.e., there is a correlation between the values in these two columns.

We have decided against the inclusion of correlation features into the domain and data table profiles because of the following reasons: First, correlations are often implicitly captured by the column profiles (e.g., in Fig. 4.1, the second column's mean value is less than the third column's mean value). Second, the variety of data types requires different correlation and dependency measures that are hard to compare. Third, we observed that the number of column pairs heavily exceeds the number of potentially meaningful correlations in our datasets. For example, consider the football

[15] http://dbpedia.org/ontology/

dataset where the length of the first names may be compared to the length of last names, team names, the number of goals, ..., potentially leading to correlations by chance. Fourth, the computation and semantic representation of all possible column combinations are impractical due to the quadratic number of pairwise comparisons.

4.7.3 Asymmetry between Domain Profiles and Data Table Profiles

The domain profile and the data table profile can vary, even though they represent the same knowledge. Consider our running example of weather observations. In the table shown in Fig. 4.1, three rows refer to the sensor labeled "S1" but only two rows refer to the other sensors. When modeled as a knowledge graph following the mapping shown in Fig. 4.4, each sensor is modeled precisely as one node in the knowledge graph. Consequently, the statistical characteristics related to the sensors vary between the data table profile and the domain profile.

4.7.4 Lightweight Semantic Dataset Profiles

Lightweight semantic profiles generated by *Tab2KG* can be utilized as a compact domain and dataset representation to complement and enrich existing dataset catalogs. Such profiles can be generated automatically from the existing datasets and described using the DCAT[16] and the SEAS[17] vocabularies to facilitate their reusability. We believe that lightweight semantic profiles presented in this article are an essential contribution that can benefit a wide range of semantic applications beyond semantic table interpretation.

4.8 Discussion

4.8.1 Discussion

In this chapter, we presented *Tab2KG* – an approach for tabular data semantification. *Tab2KG* relies on domain profiles that enrich the relations in a domain ontology and serve as a lightweight domain representation. *Tab2KG* matches these profiles with tabular data using one-shot learning. Our evaluation shows that *Tab2KG* outperforms the baselines for semantic table interpretation of five real-world datasets. In future work, we plan to consider how to integrate user feedback into the *Tab2KG* pipeline, to support an extension of the domain ontology in cases where tabular data contains previously unseen relations. Furthermore, the domain profiles generated by *Tab2KG*

[16] https://www.w3.org/TR/vocab-dcat-2/
[17] https://ci.mines-stetienne.fr/seas/index.html

can build a basis for a compact domain representation to complement and enrich dataset catalogues.

Tab2KG demonstrates that knowledge graphs do not necessarily need to be huge, general-purpose solutions that are well placed into the Linked Open Data Cloud. The representation of single tables as knowledge graphs already opens up several possibilities for making the lives of data scientists easier – with benefits including increased efficiency and robustness of data analytics workflows. By automating the step of transforming a data table to a knowledge graph, *Tab2KG* takes an important step to assisting users without expertise in ontologies and knowledge graphs and adds an important layer of abstraction.

5 Enrichment of an Event Knowledge Graph

After the creation of a knowledge graph, be it an event knowledge graph or any other kind of knowledge graph created from tabular data, there is no reason to assume it is complete. One reason being the open-world assumption under which there is no demand for a knowledge graph to be complete at any point in time, the other reason being the fact that the world is changing – and with it does the represented knowledge. The latter case is particularly relevant in the case of event knowledge graphs.

To address the problem of knowledge graph incompleteness, we proposed **RQ2** in Section 1.2, which calls for the enrichment of a knowledge graph. Considering the creation of such a knowledge graph already relied on external resources and probably subsumed all their information, we now go a step further and envision the enrichment of a knowledge graph without any further external knowledge. To this end, this chapter deals with the enrichment of an event knowledge graph based on event series.

5.1 Introduction

Event series, such as sports tournaments, music festivals and political elections are sequences of recurring events. Prominent examples include the Wimbledon Championships, the Summer Olympic Games and the United States presidential elections. The provision of reliable reference sources for event series is of crucial importance for many real-world applications, for example in the context of Digital Humanities and Web Science research [SA00, GBRD18, GD18b], as well as media analytics and digital journalism [MB16, SAMA17].

Popular knowledge graphs such as Wikidata, DBpedia and *EventKG* V1.1 cover event series only to a limited extent. This is due to multiple reasons: First, entity-centric knowledge graphs such as Wikidata and DBpedia do not sufficiently cover events and their spatio-temporal relations (see the analysis in Section 3.6.1). Second, reference sources for knowledge graphs such as Wikipedia often focus on recent and

current events to the detriment of past events [KL12]. This leads to the deficiency in supporting event-centric applications that rely on knowledge graphs.

In this chapter, we tackle a novel problem of event series completion in a knowledge graph. In particular, we address two tasks: 1) We predict missing sub-event relations between events existing in a knowledge graph, and 2) we infer real-world events that happened within a particular event series but are missing in the knowledge graph. We also infer specific properties of such inferred events such as a label, a time interval and locations, where possible. Both addressed tasks are interdependent. The prediction of sub-event relations leads to an enriched event series structure, facilitating inference of further missing events. In turn, event inference can also lead to the discovery of new sub-event relations.

The proposed *HapPenIng* approach exclusively utilises information obtained from the knowledge graph, without referring to any external sources. This characteristic makes *HapPenIng* approach unique with respect to the event inference task. In contrast, related approaches that focus on the knowledge graphs population depend on external sources (e.g., on the news [KVW14, YRH+18]).

Contributions. The contributions of this chapter include:

- A novel supervised method for sub-event relation prediction in event series.
- An event inference approach to infer real-world events missing in an event series in the knowledge graph and properties of these events.
- A dataset containing new events and relations inferred by *HapPenIng*:
  - over 5,000 events and nearly 90,000 sub-event relations for Wikidata, and
  - over 1,000 events and more than 6,000 sub-event relations for DBpedia.

Our evaluation demonstrates that the proposed *HapPenIng* approach achieves a precision of 61% for the sub-event prediction task (outperforming the state-of-the-art embedding-based baseline by 52 percentage points) and 70% for the event inference task (outperforming a naive baseline by 44 percentage points). Our dataset with new sub-event relations and inferred events is available online[1].

5.1.1 Example: Wimbledon Championships

The Wimbledon Championships (WC), a famous tennis tournament, are an *event series* that takes place in London annually since 1877. As of April 2019, Wikidata had 132 WC editions and 915 related sub-events, for example, Women's and Men's Singles and wheelchair competitions. However, according to our analysis, this event series is incomplete. In particular, the *HapPenIng* approach proposed in this chapter

[1] http://eventkg.l3s.uni-hannover.de/happening

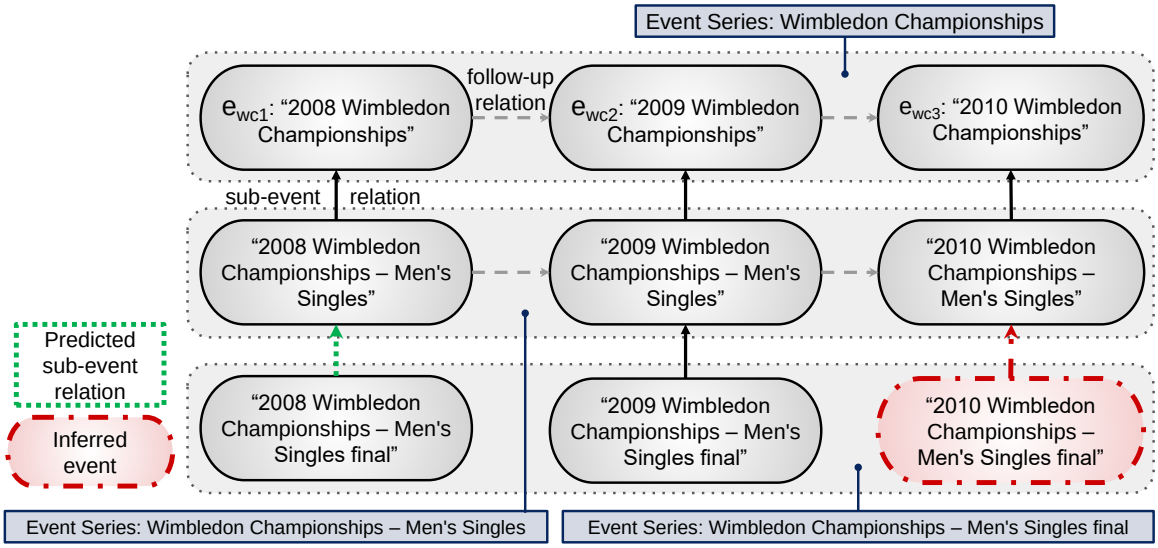


Figure 5.1. A fraction of the event graph containing the Wimbledon Championships (WC) events. Nodes represent events. Solid arrows represent sub-event relations. Dashed arrows represent follow-up event relations. The three upper events are the *WC* editions.

was able to generate 125 sub-event relations and 15 event instances related to this event series that are currently missing in Wikidata.

Figure 5.1 illustrates a small fraction of the *Event Graph* that contains event nodes and their relations as available in Wikidata as of September 18, 2018. For each year, Wikidata includes an event edition, such as the *2008 WC*. The individual competitions such as the *Men's Singles* are provided as sub-events of the corresponding edition.

In this example, we can illustrate two tasks of the event series completion tackled in this chapter: (i) Sub-event prediction: The missing sub-event relation between the Men's Singles final of 2008 and the Men's Singles competition in 2008 can be established; and (ii) Event inference: The missing event instance labelled *2010 WC — Men's Singles final* can be inferred as a sub-event of the Men's Singles 2010.

Outline. The remainder of this chapter is organised as follows: First, we give more specific background (Section 5.2) which is particularly relevant in this chapter and connect it to the methods for knowledge graph enrichment that were presented in Section 2.4. Then, we formally define the problems of sub-event prediction and event inference in Section 5.3. In Section 5.4, we describe our methods for approaching the two tasks of event series completion, followed by an evaluation of our results in Section 5.5. Finally, we discuss the findings of this chapter (Section 5.6).

5.2 Specific Background

In Section 2.4.2, we introduced three ways for knowledge graph enrichment, including link prediction and the use of external resources. In contrast to link prediction, *HapPenIng* generates new events not originally present in the knowledge graph and profits from the inclusion of textual and tempo-spatial features on top of embeddings. Instead of using external resources, *HapPenIng* solely relies on the information inherent to the knowledge graph and does not depend on the availability of the text corpora. In general, none of the knowledge graph completion and refinement tasks has yet considered the inference of new nodes given only the knowledge graph itself [WMWG17, Pau17].

As human-curated knowledge graphs such as Wikidata demand a high quality of inserted data, there have been several tools developed that help integrating automatically generated information with the respective knowledge graph. One of these tools is the Primary Sources Tool [PTVS+16], where suggestions for new relations are confirmed manually, and [DPRN17] that provides an overview of potentially missing information. Such tools can help to integrate inferred event series data into existing knowledge graphs.

5.3 Problem Statement

In this chapter, we consider an event graph, which is a knowledge graph solely containing event-related knowledge.

Definition 5.1. *An **event graph** $G_{\mathcal{V}} = (\mathcal{V}, R_S \cup R_F)$ is a knowledge graph as defined in Definition 2.1, with the following restrictions on the types of nodes and relations: The nodes of the event graph $\mathcal{V}$ represent real-world events. The edges R_S represent sub-event relations: $R_S \subseteq \mathcal{V} \times \mathcal{V}$. The edges R_F represent follow-up event relations: $R_F \subseteq \mathcal{V} \times \mathcal{V}$.*

Events in $G_{\mathcal{V}}$ represent real-world happenings; the key properties of an event in the context of event series include an event identifier, an event label, a happening time interval and relevant locations.

Definition 5.2. *Given an event graph $G_{\mathcal{V}} = (\mathcal{V}, R_S \cup R_F)$, an **event** $e \in \mathcal{V}$ is something of societal importance that happened in the real world (see Definition 2.2). e is represented as a tuple $e = \langle uri, l, t, L \rangle$, where uri is an event identifier, l is an event label, $t = \langle t_s, t_e \rangle$ is the happening time interval with t_s, t_e being its start and end time. L is the set of event locations.*

An event can have multiple sub-events. For example, the *WC Men's single final 2009* is a sub-event of *2009 WC*.

Definition 5.3. *An event $e_s \in \mathcal{V}$ is a **sub-event** of the event $e_p \in \mathcal{V}$, i.e., $(e_s, e_p) \in R_S$, if e_s and e_p are topically related and e_s is narrower in scope. R_S is the set of sub-event relations.*

We refer to e_p as a parent event of e_s. Typically, e_s happens in the temporal and geographical proximity of e_p.

An event can be a part of an event series. An example of an event series are the *WC* that have the *2008 WC* as one of its editions.

Definition 5.4. *An **event series** $s = \langle e_1, e_2, \ldots, e_n \rangle$, $\forall e_i \in s : e_i \in \mathcal{V}$, is a sequence of topically related events that repeatedly occur in a similar form. The sequence elements are ordered by the event start time and are called **editions**. We refer to the set of event series as S.*

The follow-up relations R_F connect event editions within an event series. For example, the *2009 WC* is the follow-up event of the *2008 WC*.

Definition 5.5. *Given an event series $s = \langle e_1, e_2, \ldots, e_n \rangle$, e_j is a **follow-up event** of e_i, i.e., $(e_i, e_j) \in R_F$, if $e_i \in s$ and $e_j \in s$ are the neighbour editions in s and e_i precedes e_j. R_F is the set of follow-up relations.*

The sub-event relations in an event graph are often incomplete, as a consequence of the open-world assumption. In particular, we denote the set of real-world sub-event relations not included in the event graph as R_S^+. Then the task of sub-event prediction can be defined as follows:

Definition 5.6. *Given an event graph $G_{\mathcal{V}} = (\mathcal{V}, R_S \cup R_F)$ and two events $e_s \in \mathcal{V}$ and $e_p \in \mathcal{V}$, the task of the **sub-event prediction** is to decide if e_s is a sub-event of e_p, i.e., to determine if $(e_s, e_p) \in R_S \cup R_S^+$, where R_S^+ is a set of real-world sub-event relations not included in the event graph.*

The set of real-world event representations included in an event graph is often incomplete as well. The context of event series can help to infer real-world events missing in particular editions.

Definition 5.7. *Given an event graph $G_{\mathcal{V}} = (\mathcal{V}, R_S \cup R_F)$ and an event series $s = \langle e_1, e_2, \ldots, e_n \rangle$, with $e_1, e_2, ..., e_n \in \mathcal{V}$, the task of **event inference** is to identify a real-world event $e_f \in \mathcal{V} \cup \mathcal{V}^+$ that belongs to the series s. Here, $\mathcal{V}^+$ is a set of real-world events that are not included in the event graph. In particular, e_f is a sub-event of the edition $e_i \in s$, i.e., $(e_f, e_i) \in R_S \cup R_S^+$.*

5.4 *HapPenIng*: Approach

We address event series completion in two steps: First, we adopt a classification method to predict sub-event relations among event pairs. Second, we develop a graph-based approach to infer events missing in particular editions through event series analysis. A pipeline of the overall approach is shown in Figure 5.2.

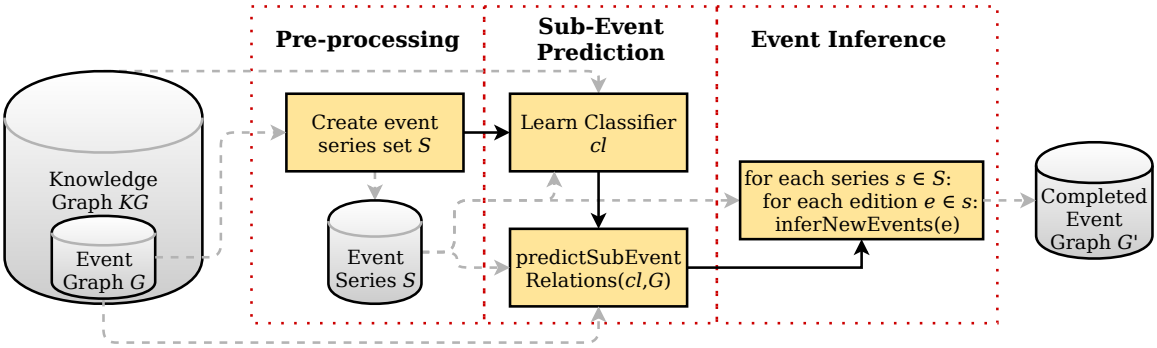


Figure 5.2. The *HapPenIng* pipeline. Solid arrows represent the processing order. Dashed arrows represent the data flow.

5.4.1 Sub-Event Prediction

We model the problem of sub-event prediction as a classification problem. Given an event pair (e_s, e_p), we aim to predict whether e_s is a sub-event of e_p:

$$sub-event(e_s, e_p) = \begin{cases} true, & \text{if}(e_s, e_p) \in R_S \cup R_S^+; \\ false, & \text{otherwise.} \end{cases} \tag{5.1}$$

Features

We adopt textual, spatio-temporal and embeddings features.

Textual features (`TEX`): Events connected through a sub-event relation can have similar or overlapping labels whose similarity is measured using textual features. Such features are also applied on the *template labels*. Template labels are series labels obtained from the original event labels after removal of any digits. The textual features we consider include:

- Label Containment: 1 if $e_p.l$ is a sub-string of $e_s.l$, 0 otherwise.
- LCS Fraction: The length of the Longest Common Sub-string (LCS) of $e_s.l$ and $e_p.l$, compared to the shorter label: $f_{\text{LCS Fraction}}(e_s, e_p) = \frac{LCS(e_s.l, e_p.l)}{min(|e_s.l|, |e_p.l|)}$.
- Unigram Similarity: The labels of both events are split into word unigrams. The feature value is the Jaccard similarity between the unigram sets: $f_{\text{Unigram Similarity}}(e_s, e_p) = \frac{\text{unigrams}(e_s.l) \cap \text{unigrams}(e_p.l)}{\text{unigrams}(e_s.l) \cup \text{unigrams}(e_p.l)}$.
- Template Containment, Template LCS Fraction, Template Unigram Similarity: These features are computed equivalent to the label features but are based on the template labels.

- Label Cosine Similarity: The cosine similarity between event labels based on tf-idf vectors to take frequency and selectivity of terms into account.
- Parent Event Label Length: $f_{\text{Parent Event Label Length}}(e_s, e_p) = |e_p.l|$.
- Sub-Event Label Length: $f_{\text{Sub-Event Label Length}}(e_s, e_p) = |e_s.l|$.

Spatio-temporal features (`STP`): We assume that sub-events happen in the temporal proximity of their parent events. We consider the temporal proximity through temporal overlap, containment and equality.

- Time Overlap: 1 if $e_s.t \cap e_p.t \neq \varnothing$, 0 otherwise.
- Time Containment: 1 if $e_s.t \subseteq e_p.t$, 0 otherwise.
- Time Equality: 1 if $e_s.t = e_p.t$, 0 otherwise.

Sub-events typically happen in the geographical proximity of their parent events. Therefore, we introduce Location Overlap - a spatial feature that assigns a higher score to the event pairs that share locations:

- Location Overlap: 1 if $e_s.L \cap e_p.L \neq \emptyset$, 0 otherwise.

Embedding features (`EMB`): The link structure of the knowledge graph can be expected to provide important insights into possible event relations. First, we can expect that this structure provides useful hints towards predicting sub-event relations, e.g., follow-up events can be expected to have a common parent event. Second, events related to different topical domains (e.g., politics vs sports) are unlikely to be related through a sub-event relation. To make use of this intuition, we train an embedding on the knowledge graph using any relations connecting two events in E. For this feature, we pre-train the embeddings following the STransE embedding model [NSQJ16], which provides two relation-specific matrices $\boldsymbol{W_1}$ and $\boldsymbol{W_2}$, a relation vector $\boldsymbol{r}$ and entity vectors (here, $\boldsymbol{e_s}$ and $\boldsymbol{e_p}$). Intuitively, given that model, we can compare the embedding of an event with the embedding of the assumed parent event plus the embedding of the sub-event relation (sE):

- Embedding Score: $f_{\text{Embedding}}(e_s, e_p) = \|\boldsymbol{W}_{r_{\text{sE}},1}\boldsymbol{e}_p + \boldsymbol{r}_{\text{sE}} - \boldsymbol{W}_{r_{\text{sE}},2}\boldsymbol{e}_s\|_{\ell_1}$

Training the sub-event classifier

To train a classifier given the features presented above, a set of labelled event pairs is required. The set of positive examples contains all event pairs with known sub-event relations in the event graph $G_{\mathcal{V}}$. Formally, given the set $\mathcal{V}$ of events, this is the set $C_+ = \{(e_s, e_p) | (e_s, e_p) \in R_S\}, e_s \in \mathcal{V}, e_p \in \mathcal{V}$.

In addition, a set of negative examples, i.e., event pairs without sub-event relation is required. When composing event pairs randomly, most of the paired events would be highly different (e.g., having highly dissimilar labels and no spatio-temporal overlap). Consequently, the model would only learn to distinguish the most straightforward cases. To address this problem, we collect a set of negative examples C_- that has as many event pairs as C_+, and consists of four equally-sized subsets with the following condition for each contained event pair (e_s, e_p), where $(e_s, e_p) \notin R_S$:

- Both events are from the same event series, but $(e_s, e_p) \notin R_S$. Example: (*1997 WC — Women's Doubles*, *2009 WC — Men's Singles final*).
- Both events have the same parent event. Example: (*2009 WC — Men's Singles*, *2009 WC — Women's Singles*).
- The parent of e_s's parent is the same as e_p's parent. Example: (*2009 WC — Men's Singles final*, *2009 WC — Women's Singles*).
- e_s is a transitive, but not a direct sub-event of e_p. Example: (*2009 WC — Men's Singles final*, *2009 WC*).

Note that we only consider direct sub-event relations to be valid positive examples. In particular, we aim to learn to distinguish the directly connected sub-events from transitive relations, as well as to distinguish similar events that belong to different editions. Due to the inherent incompleteness of the event graph under the open-world assumption, a missing sub-event relation does not necessarily imply that this relation does not hold in the real world. However, we expect that false-negative examples would occur only rarely in the training set, such that the resulting model will not be substantially affected by such cases.

Overall, the set of training and test instances C contains all positive sub-event examples C_+ found in the event graph, and an equally sized set of negative examples C_- that consists of the four event pair sets described above.

Predicting sub-event relations using the classifier

The trained classifier is adopted to predict missing sub-event relations within event series. We apply an iterative algorithm, given a classifier *cl* and the event graph $G_{\mathcal{V}}$. As it is not feasible to conduct a pairwise comparison of all events in $G_{\mathcal{V}}$, we limit the number of events compared with their potential parent event: For each potential parent event e_p that is part of an event series, a set of candidate sub-events is selected as the set of events with the largest term overlap with the potential parent event label. For each candidate event, the classifier *cl* predicts whether this event is a sub-event of e_p. To facilitate the prediction of sub-event relations in cases where the parent event initially is not a part of the series, the procedure is run iteratively until no new sub-event relations are found.

5.4.2 Event Inference

The task of event inference is to infer real-world events not initially contained in the event graph (i.e., events in the set $\mathcal{V}^+$). We infer such missing events and automatically generate their key properties such as label, time frame and location, where possible. The intuition behind event inference is that the event graph indicates specific patterns repeated across editions. Thus, we approach this task via comparison of different editions of the same event series to recognise such patterns. Consider the WC example in Figure 5.1: Although there is no event instance for the *2010 Men's Singles final*, we can infer such instance from the previous edition *2009 Men's Singles final*.

Event Series Pre-processing

We pre-process the set S of event series to avoid cycles or undesired dependencies within the single series. Each event series is transformed into a sequence of acyclic rooted trees where each root represents one particular edition of the series. Events or relations violating that structure are removed from the series. If removal is not possible, we exclude such series from S.

An important concept of the event inference is the concept of a sub-series: A series s_p has a sub-series s_s if the sub-series contains sub-events of s_p. For example, the *WC — Men's Singles final* series is a sub-series of the *WC — Men's Singles* because the event *2009 WC — Men's Singles final* is a sub-event of *2009 WC — Men's Singles*. We determine sub-series relation as:

Definition 5.8. *An event series $s_s \in S$ is a **sub-series** of $s_p \in S$ if for an event $e_p \in s_p$ there is a sub-event in s_s: $\exists(e_s, e_p) \in R_S : e_p \in s_p \wedge e_s \in s_s$.*

Inferring New Events

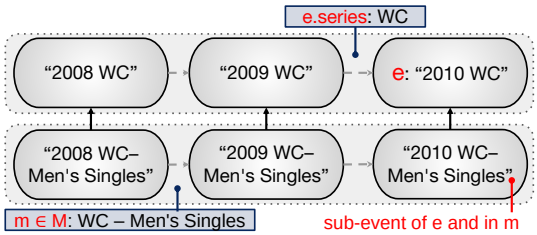
The intuition behind event inference is to identify similar patterns in the different editions of an event series. According to Definition 5.4, the editions of an event series repeatedly occur in a similar form. This way, events repeated in most of the editions of the series but missing in a particular edition can be inferred. To do so, we process all editions in the event graph and inspect whether its neighboured editions have a sub-event not covered in the particular edition.

Algorithm 1 illustrates our event inference approach. As shown in our pipeline (Figure 5.2), this algorithm is invoked for each edition e of the event series in S. First, a set M is constructed that contains all sub-series of the current edition's series, i.e., $e.series$ (line 2). Then, the algorithm removes all series from M, for which the current edition contains events already (line 4). That way, M is reduced to a set of event series not covered by the sub-events of the current edition e.

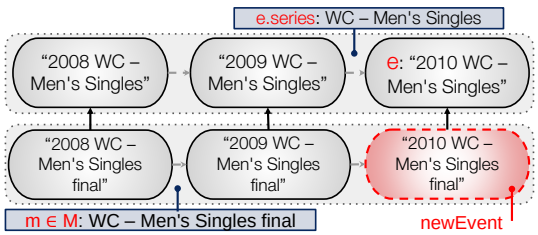
For each remaining sub-series $m \in M$, a new event is inferred that is a sub-event of the current edition e and a part of m. Within the respective method

Algorithm 1 Event Inference

```
1: procedure InferSubEvents(e)
2:     M ← getSubSeries(e.series)
3:     for each e_s ∈ {e_s|(e_s, e) ∈ R_S} do
4:         M = M \ e_s.series
5:     for each m ∈ M do
6:         if constraintsNotSatisfied(m, e) then
7:             continue
8:         newEvent ← inferEvent(e, m)
9:         if oldEvent ← findEvent(V, newEvent.l) ≠ ∅ then
10:            R_S ← R_S ∪ (e, oldEvent)
11:        else
12:            V ← V ∪ newEvent; R_S ← R_S ∪ (e, newEvent)
13:    for each e_s ∈ {e_s|(e_s, e_p) ∈ R_S} do
14:        inferSubEvents(e_s)
```



(a) Step 1: The event inference algorithm is invoked with the *2010 WC* event e. For the sub-series m of $e.series$, *2010 WC — Men's Singles* is already a sub-event of e. No new event is inferred.



(b) Step 2: The algorithm is now invoked with the *WC 2010 — Men's Singles* event e. For the sub-series m of $e.series$, there is no sub-event of e. A new event is inferred.

Figure 5.3. Event inference example for the Wimbledon Championships.

`inferEvent`(e, M), a new label, time span and set of locations is generated as described later. The algorithm is invoked recursively with all known (also newly identified) sub-events. To increase precision, a sub-series m is only retained in M if a set of constraints is satisfied (line 6). These constraints are described later in this section.

The event inference algorithm can infer an event for which an equivalent event already exists in the event graph. To avoid the generation of such duplicate events, we check if an event with the same label as the newly inferred event exists in the event graph. In this case, the algorithm adds a new sub-event relation across the existing events to the event graph and discards the inferred event (line 10).

Algorithm 2 Label Generation

```
1: procedure GenerateLabel(e, m)
2:     mostSimilarEvents ← getSimilarEvents(e, e.series)
3:     sortEventsByEditionCloseness(e, mostSimilarEvents)
4:     c ← mostSimilarEvents[0]
5:     c' ← c', s.t. (c', c) ∈ R_S ∧ c' ∈ m
6:     l ← ""; r ← c'.l; δ_prev ← ∅
7:     for each δ ∈ getEdits(c.l, e.l) do
8:         if δ.op = DELETE then
9:             δ_prev ← δ
10:        else if δ.op = INSERT ∧ δ_prev.op = DELETE then
11:            l ← l + r[: r.indexOf(δ_prev.text)] + δ.text
12:            r ← l + r[r.indexOf(δ_prev.text) + len(δ_prev.text) :]
13:        else if not (δ.op = EQUAL ∧ δ_prev = ∅) then
14:            return ∅
       return l + r
```

Table 5.1. Generating the label *2010 WC - Men's Singles*. The edit operations δ are the result of Myers' algorithm to detect the edit operations between *2009 WC - Men's Singles* and *2010 WC - Men's Singles*. The final label is the concatenation of l and r.

step	δ.op	δ_{prev}.op	δ.text	l	r
init					2009 WC - Men's Singles final
1	DELETE		2009		2009 WC - Men's Singles final
2	INSERT	DELETE	2010	2010	WC - Men's Singles final
3	EQUAL		WC - Men's Singles	2010	WC - Men's Singles final

Wimbledon Championships Example

Consider the example in Figure 5.1, with the goal to infer new events within the edition e_{wc_3}: *2010 WC*. Figure 5.3a depicts the first step when invoking the algorithm `InferSubEvents(`e_{wc_3}`)` (without constraints). The edition becomes the input event e, and its series $e.series$ is *WC*. The event series *WC — Men's Singles* (m) is identified as one of its immediate sub-series in M. However, as there is already an event *2010 WC — Men's Singles* that is a sub-event of e and part of that sub-series m, it is removed from M. Therefore, M is empty and no new events are inferred at this point.

Subsequently, Algorithm 1 is executed with the sub-event *2010 WC — Men's Singles* as input edition e, as shown in Figure 5.3b. Here, the sub-series is *WC — Men's Singles final* which is inserted in M. Consequently, a new event is created that is a sub-event of e and part of the event series *WC — Men's Singles final*.

Label Generation

Each newly generated event requires a label. This label is generated by exploiting the labels within its event series, as shown in Algorithm 2. The input is its future parent event e and its event series m. First, the events in the parent series $e.series$, whose labels are most similar to the label of e, are collected (line 2). Then, within this set of events, the one from the closest event edition and its sub-event in m is selected (lines 3 - 5). Finally, the label of that event is transformed into the new label by applying the same edit operations δ (i.e., equality, delete or insert) as if we transformed the parent event labels (lines 6 - 14). To identify the edits, we adopt the difference algorithm by Myers [Mye86].

Example: Consider the newly added event in Figure 5.3b. As an input to the algorithm, there is e which is the event *2010 WC — Men's Singles* and the series m consisting of the Men's Singles finals of 2008 and 2009. First, the event *2009 WC — Men's Singles* within $e.series$ is identified as the most similar event c. c' is the sub-event of c that is also in m: *2009 WC — Men's Singles final*. Given $c'.l$ and the edit operations δ between the labels of e and c, Table 5.1 shows how they are used to generate the correct label *2010 WC — Men's Singles final*.

Location and Time Generation

Each event can be assigned a happening time and a set of locations. In both cases, we use a rule-based approach.

Locations: Some events, such as the Olympic Games change their location with every edition. Currently, we reconstruct event locations only if they remain unchanged across editions: If there is a location assigned to every event $s \in m$, this location is also assigned to e. In future work, we intend to utilise sub-location relations, that facilitate the generation of correct locations at a lower level of geographical granularity.

Happening Times: Three rules are applied in the following order until a happening time is identified:

1. If the happening time of each event $s \in m$ equals its parent event's happening time, also e adopts its happening time directly from its parent event.

2. If the happening time of each event $s \in m$ is modelled as a whole year, the happening time of e is also modelled as the same year as any of its (transitive) parent events.

3. If the event label contains a year expression, that part is transformed into its happening time.

Constraints

We propose several configurations of constraints to decide whether an event should be created:

- Baseline (`BSL`): No constraints.
- Time Evolution (`EVO`): The constraints are only satisfied if there was at least one event in the series that happened before e. For example, the *Wimbledon Women's Doubles* was held for the first time in 1913, so it would be wrong to generate an event for the *Women's Doubles* series in 1912 and before.
- Interval (`INT`): The constraints are only satisfied if there was at least one event in the series that happened before and at least one event in the series that happened after e. Under this constraint, events that re-occurred only until a specific edition are not generated for each edition. An example is the tug of war which was part of only six Olympic Summer Games.
- Window (`WIN`): Given start and end thresholds a and b, this constraint is satisfied if there is at least one event within the last a editions of the series that happened before e and at least one event in the following b editions that happened after e. For example, Tennis competitions in the Olympic Summer Games were held between 1896 and 1924, and then only since 1984. The Window constraint helps to identify such gaps.
- Coverage (`COV`): Event series are only valid if they are part of a sufficient fraction of the editions: $|m|/|S| \geq \alpha$, given a threshold α.
- Coverage Window (`CWI`): A combination of `WIN` and `COV`: The coverage is only computed after restricting both event series to the dynamic time window.
- Evolution Coverage Window (`ECW`): A combination of `EVO`, `WIN` and `COV`: The coverage is only computed after restricting both event series to the dynamic time window, and if at least one event in the series happened before e.

5.5 Evaluation

The goal of the evaluation is to assess the performance of the *HapPenIng* approach with respect to the sub-event prediction and event inference tasks.

5.5.1 Data Collection and Event Graph Construction

We run our experiments on event graphs extracted from two sources: (i) Wikidata as of October 25, 2018 (*Wikidata Event Graph*), and (ii) DBpedia from the October

2016 dump (*DBpedia Event Graph*). Both datasets are enriched with additional information regarding events obtained from the *EventKG* knowledge graph [GD18a]. Compared to other knowledge graphs, *EventKG* contains more detailed information regarding the spatio-temporal characteristics of events (see the analysis in Section 3.6.1). More concretely, events in the event graph are enriched with location and time information using the properties *sem:hasPlace*, *sem:hasBeginTimeStamp* and *sem:hasEndTimeStamp* of *EventKG*.

One event graph containing events, sub-event relations and follow-up relations, as well as a set S of event series is constructed for each dataset. For the *Wikidata Event Graph*, we collect as events all data items that are (transitive) instances of the "event" class[2]. Event series are extracted using the `instance of`[3] and the `series`[4] properties in Wikidata. For the *DBpedia Event Graph*, we extract events using the `dbo:Event` class and series assignments using the provided Wikipedia categories. In both cases, we apply two heuristics to ensure that only event series compatible with Definition 5.4 are extracted: (i) We only consider series with mostly homogeneous editions. To this end, we make use of the Gini index [RS04], a standard measure for measuring impurity. In our context, it is used to assess the diversity of the template labels of editions in an event series. We reject the (rare) cases of event series with high Gini impurity, where the edition labels do not follow any common pattern.[5] An event is kept in S if the set of template labels of its editions shows a Gini impurity less than 0.9. Besides, we ignore editions whose removal decreases that impurity. (ii) We ignore events typed as military conflicts and natural disasters because such events typically do not follow any regularity. If we can find connected sub-graphs of events in the event graph through sub-event and follow-up relations, but the data item representing that series is missing in the dataset, we add a new unlabelled event series to S. To train the embeddings, we collect all relations connected to events.

The extraction process results in a *Wikidata Event Graph* $G_{\mathcal{V}}$ containing $|\mathcal{V}| = 352,235$ events (*DBpedia Event Graph*: $92,523$) and $|S| = 9,007$ event series (*DBpedia Event Graph*: $1,871$). As input to train the embeddings, there are $279,004,908$ relations in Wikidata and $18,328,678$ relations in DBpedia. Both event graphs, as well as embeddings, annotated samples and other evaluation datasets described in the remainder of this section, are available online.[6]

[2] https://www.wikidata.org/wiki/Q1656682
[3] https://www.wikidata.org/wiki/Property:P31
[4] https://www.wikidata.org/wiki/Property:P179
[5] For example, the event series "TED talk", whose set of edition template labels (e.g., "Avi Reichental: What's next in 3D printing" and "Amanda Palmer: The art of asking") has a high Gini impurity, is not included in the set of event series.
[6] http://eventkg.l3s.uni-hannover.de/happening

Table 5.2. 10-fold cross-validation of the sub-event prediction using different classifiers and all the introduced features. `STransE` is the baseline we compare to.

Method		Wikidata TP	Wikidata TN	Wikidata FP	Wikidata FN	Wikidata Accuracy	DBpedia Accuracy
Baseline	`STransE`	46,479	43,143	6,949	13,859	0.81	0.50
HapPenIng configurations	`LOG`	54,345	46,605	3,487	5,993	0.91	0.87
	`SVM`	55,958	48,825	1,267	4,380	0.95	0.92
	`RF`	**58,649**	**49,497**	**595**	**1,689**	**0.98**	**0.97**

5.5.2 Sub-Event Prediction

Training and Test Set Generation

Before running the experiments, a set of positive and negative sub-event relations is created from the event graphs as described in Section 5.4.1. In total, this collection of relations consists of $55,217$ event pairs within S that were extracted as correct sub-event pairs from Wikidata (DBpedia: $16,763$) and the same number of negative event pairs.[7] This collection is split into ten folds to allow 10-fold cross-validation. We learn the STransE embeddings as described in Section 5.4.1 for each fold, with its parameters set as follows: SGD learning rate $\lambda = 0.0001$, the margin hyper-parameter $\gamma = 1$, vector size $k = 100$ and $1,000$ epochs. While learning the embeddings on the folds, we exclude the sub-event relations from the respective test set.

Baseline

As a baseline for sub-event prediction, we utilise an embedding-based link prediction model based on the STransE embeddings [NSQJ16]. Given an input event, this model retrieves a ranked list of candidate sub-events with the corresponding scores. We use these scores to build a logistic regression classifier. STransE is a state-of-the-art approach that had been shown to outperform previous embedding models for the link prediction task on the FB15K benchmark [BUGD+13].

Classifier Evaluation

Table 5.2 shows the results of the 10-fold cross-validation for the sub-event prediction task, with three different classifiers: `LOG` (Logistic Regression), `RF` (Random Forest) and `SVM` (Support Vector Machine with linear kernel and normalisation) in terms of classification accuracy ($\frac{TP+TN}{TP+TN+FP+FN}$, where TP are true positives, TN true negatives, FP false positives and FN false negatives). Among our classifiers, the `RF`

[7] Existing benchmark datasets do not contain a sufficient amount of sub-event relations. For example, FB15K [BUGD+13] only contains 224 triples containing one of the Freebase predicates */time/event/includes_event*, */time/event/included_in_event* or */time/event/instance_of_recurring_event*.

Table 5.3. 10-fold cross-validation of the sub-event prediction using the RF classifier for Wikidata and DBpedia with different feature sets.

Feature Group	Wikidata Accuracy	DBpedia Accuracy
All features: TEX, STP, EMB	**0.98**	0.97
No spatio-temp. features: TEX, EMB	0.97	0.96
No textual features: STP, EMB	0.82	0.73
No embedding: TEX, STP	0.98	**0.97**

classifier performs best, with an accuracy of nearly 0.98 in the case of the *Wikidata Event Graph* and 0.97 for the *DBpedia Event Graph*. The results show a clear improvement over the STransE baseline, outperforming the baseline by more than 16 percentage points in case of the RF classifier for Wikidata. For DBpedia, the STransE baseline is outperformed by a larger margin using our proposed features. This can be explained by the insufficient number of relations for training the embeddings in DBpedia.

Table 5.3 shows the performance of the RF classifier under cross-validation with different feature groups. The combination of all features leads to the best performance in terms of accuracy. Although the use of textual features already leads to high accuracy (0.97), embedding features and spatio-temporal features help to further increase accuracy in the case of Wikidata (0.98). Again, while DBpedia does profit from the spatio-temporal features, there is no improvement when using embeddings, due to the insufficient data size.

Wikidata Statistics and Examples

While the classifiers demonstrate very accurate results on the test sets, the performance on predicting sub-event relations not yet contained in G requires a separate evaluation. As explained in Section 5.4.1, a large number of predictions is needed that could potentially also lead to a large number of false positives, even given a highly accurate classifier. The actual label distribution is skewed towards unrelated events, and we are now only classifying event pairs not yet contained in R_S. In fact, running the sub-event prediction algorithm using the best-performing RF classifier with all features leads to the prediction of $85,805$ new sub-event relations not yet contained in Wikidata and $5,651$ new sub-event relations in DBpedia.

To assess the quality of the predicted sub-event relations that are not initially contained in R_S, we extracted a random sample of 100 sub-event relations consisting of an event and its predicted sub-event and manually annotated each pair as correct or incorrect sub-event relation. According to this manual annotation, 61% of the sub-event relations predicted with our *HapPenIng* approach that are not yet contained

Table 5.4. Complementing corrupted event series. For each corruption factor (i.e., % of removed events), we report the percentage of events that could be reconstructed.

		Wikidata			DBpedia		
		Corruption Factor					
Constraints		**5%**	**10%**	**15%**	**5%**	**10%**	**15%**
Baseline	BSL	61.81	63.13	61.83	39.58	38.40	38.17
	EVO	53,63	54.70	53.12	31.04	31.32	30.12
	INT	46.68	47.89	46.39	24.58	24.04	23.46
HapPenIng	WIN	46.06	47.45	45.94	22.71	22.27	21.93
configurations	COV	45.49	45.65	43.64	11.46	11.03	9.30
	CWI	53.36	53.93	51.32	23.96	21.96	19.43
	ECW	48.89	49.17	47.03	21.67	20.71	18.18

in the event graph correctly represent real-world sub-event relations in Wikidata (DBpedia: 42%). In comparison, the STransE baseline predicted only 46,807 new sub-event relations, and only 9% of them are correct based on manual annotation of a random 100 relations sample (DBpedia: 2%). The difference in performance on the test set and on the predicted sub-event relations not contained in R_S can be explained by the large class disbalance in the set of relations collected in the sub-event prediction procedure, such that the majority of the candidate relations are negative examples.

5.5.3 Event Inference Performance

We evaluate the event inference performance in two steps: First, we conduct an automated evaluation of recall by reconstruction of corrupted event series. Second, we assess precision by annotating random samples of new events.

Complementing Corrupted Event Series (Recall)

To evaluate the recall of the event series completion, we remove events from the event series and investigate to which extent our event graph completion constraints can reconstruct them (we consider the naive unconstrained approach BSL as our baseline). To this end, we randomly remove leaf nodes (events without sub-events) from the whole set of event series S until a specific percentage (determined by the *corruption factor*) of leaf nodes is removed. For the *Wikidata Event Graph*, there are 45,203 such leaf events in total before corruption, for DBpedia 9,600. Table 5.4 shows the results for three corruption factors (5%, 10% and 15%) and the constraints introduced in Section 5.4.2 (we set the parameters to $a = b = 5$ and $\alpha = 0.5$). As expected, the unconstrained naive approach BSL results in the highest percentage of correctly reconstructed events: More than 60% of the Wikidata and nearly 40% of the DBpedia events can be recovered, including their correct labels. If applying constraints, fewer

Table 5.5. Manual evaluation of the correctness of inferred events. For the baseline, each *HapPenIng* constraint and event graph, 100 inferred events were randomly sampled and judged as correct or not. The number of additional sub-event relations found during the event inference process is reported as well (P: Precision).

Constraints		Wikidata Inferred Events Number	P	Wikidata Relations	DBpedia Inferred Events Number	P	DBpedia Relations
Baseline	BSL	**114,077**	0.26	**16,877**	**31,410**	0.24	**3,420**
HapPenIng configurations	EVO	28,846	0.47	10,045	11,295	0.35	1,170
	INT	5,256	0.57	5,376	2,115	0.67	3,419
	WIN	3,363	0.56	4,547	936	**0.71**	783
	COV	7,297	0.54	2,712	1,313	0.45	417
	CWI	7,965	0.59	4,442	1,965	0.61	718
	ECW	5,010	**0.70**	3,687	1,364	0.70	655

events are reconstructed. In particular, the WIN constraint results in the lowest recall, as it demands to cover the event before and after the series edition within 5 editions.

Overall, we observe that *HapPenIng* is able to reconstruct more than 60% of missing events from a knowledge graph and correctly infer event labels.

Manual Assessment (Precision)

To access precision, we created random samples of 100 newly inferred events for each of the constraints proposed in Section 5.4.2 and both event graphs, and manually annotated their correctness. Table 5.5 provides an overview of the results. While the naive unconstrained approach results in a precision of less than 0.30 for both event graphs, the inclusion of constraints leads to clear improvement, with a precision of up to 0.70 for the ECW constraint for Wikidata and 0.71 for the WIN constraint for DBpedia. Table 5.5 also reports the number of additional sub-event relations created during the event inference procedure when checking for duplicate events.

Additional Statistics

The manual assessment shows that *HapPenIng* with the ECW constraints is able to infer $5,010$ new events with a precision of 70% in Wikidata and $1,364$ new DBpedia events with similar precision. Events are inferred wrongly in cases where sub-events are happening in an irregular manner. For example, this includes the wrongly inferred event "1985 Australian Open – Mixed Doubles" that was extracted although there were no Mixed Doubles in that event series between 1970 and 1985 or competitions like the men's single scull in the World Rowing Championships that used to follow a highly irregular schedule. In future, external knowledge can be used to verify the inferred

events. Differences between the Wikidata and the DBpedia results can be explained by the less complete event type assignments and the lack of a proper sub-event relation in DBpedia, where we use category assignments instead.

As the `ECW` constraint is most precise for the *Wikidata Event Graph*, we provide more insights for this constraint and event graph in the following:

- Impact of the sub-event prediction on the event inference: If the sub-event prediction step is skipped, only 3,558 new events are inferred, compared to 5,010 events otherwise.
- Additional relations: 3,687 new sub-event relations were created during the event inference step in addition to the 85,805 sub-event relations from the sub-event prediction step (in total: 89,492 new sub-event relations).
- Happening times: 99.36% of the inferred events are assigned a happening time. 0.38% of them were inferred by the first, 81.52% by the second and 18.10% by the third rule from Section 5.4.2.
- Locations: Only 79 of the 5,010 inferred events were assigned a location under the strict conditions proposed in Section 5.4.2.

Overall, the two steps sub-event prediction and event inference enable *HapPenIng* to generate ten thousands of new sub-event relations and events. These relations and new instances can be given as a suggestion to be inserted in the respective dataset using human confirmation with external tools, such as the Primary Sources Tool for Wikidata [PTVS+16].

5.6 Discussion

In this chapter, we addressed a novel problem of event series completion in a knowledge graph. The proposed *HapPenIng* approach predicts sub-event relations and real-world events missing in the knowledge graph and does not require any external sources. Our evaluation on Wikidata and DBpedia datasets shows that *HapPenIng* predicts nearly 90,000 sub-event relations missing in Wikidata (in DBpedia: over 6,000), clearly outperforming the embedding-based baseline by more than 50 percentage points, and infers over 5,000 new events (in DBpedia: over 1,300) with a precision of 70%. Our dataset was made publicly available to encourage further research.

These events and relations can be used as valuable suggestions for insertion in Wikidata and DBpedia. Manual verification is still required to conform to the goal of having a high precise event knowledge graph, as even event series can be unpredictable: For example, back in 2019, there was no way for to predict the coronavirus pandemic in 2020, which lead to the cancellation of many planned events.

HapPenIng demonstrates that under specific conditions, event knowledge graphs can be enriched not only with new edges, but with new nodes as well. *HapPenIng* may motivate researchers to think of new methods for knowledge graph enrichment. The presented approach could be extended to further resource types beyond event series, like series of cars or books. Within event knowledge graphs, there is much potential of inferring events from given relations: Consider the marriage relation between two persons, which does implicitly represent up to two events: the wedding and, optionally, the divorce. Based on these thoughts, we conclude that *HapPenIng* is an excellent example for getting more out of a knowledge graph, using strategies that go beyond the standard methods of knowledge graph enrichment presented in Section 2.4.2.

6 Application of an Event Knowledge Graph

Even the existence of a perfectly created and enriched knowledge graph does not per se lead to the access to its knowledge. As already insinuated with **RQ3**, users who are not familiar with the concept of knowledge graphs can not interact with the represented knowledge, given the lack of SPARQL expertise and the missing knowledge of the knowledge graph specific schema. To overcome this issue, we hereby introduce a chapter which is entirely based on the application of event knowledge graphs.

6.1 Introduction

Wikipedia, with more than one million articles dedicated to famous people, as well as other encyclopedic or biographical corpora on the Web, are rich sources of biographical information. These sources can help to answer questions like *"What were the notable accomplishments in the life of Barack Obama?"*, and to learn about the life of people of public interest. Researchers who analyse event-centric cross-lingual information (in particular, computer scientists, information designers, and sociologists) prefer to approach such questions by exploiting concise representations, rather than by close reading of lengthy articles [GBRD18].

Knowledge graphs can serve as a valuable resource for answering these kinds of information needs. However, a popular entity such as an influential person, a city or a large organisation can impose hundreds of temporal relations within a temporal knowledge graph. For example, the entity Barack Obama possesses 2,608 temporal relations in *EventKG*. Identifying the most important temporal relations within the temporal knowledge graph to provide a concise overview of a given entity becomes a challenging task in these settings.

Timelines are an effective method to provide a visual overview of entity-centric temporal information, such as temporal relations in a knowledge graph [ADM+15]. In particular, biography timelines describe significant happenings in a person's life

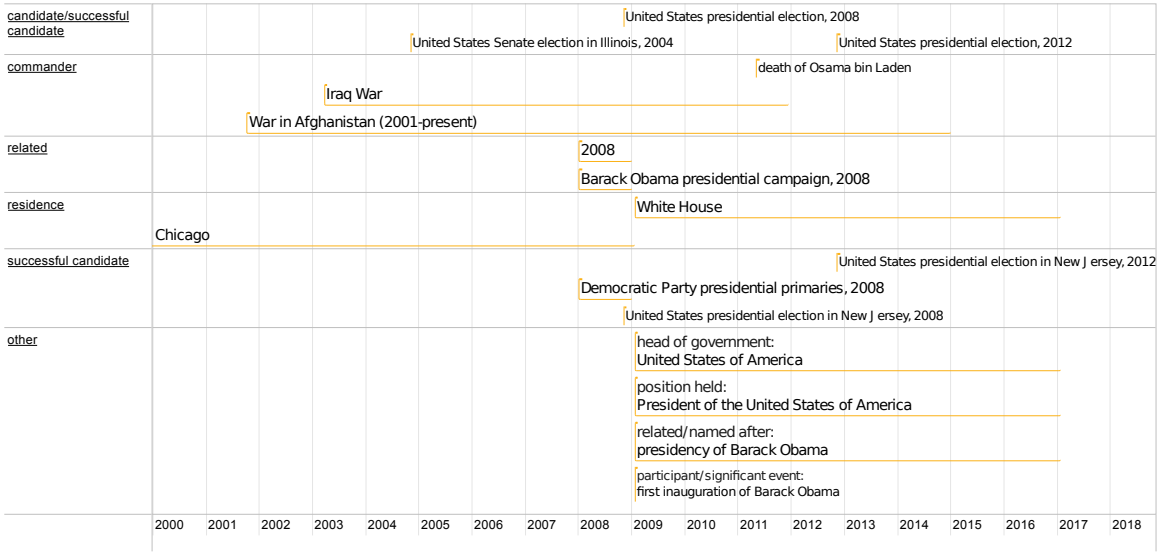


Figure 6.1. An excerpt of the biography timeline for the entity Barack Obama, generated from the *EventKG* knowledge graph using a proposed model trained on the Wikipedia abstracts of other entities. Orange lines represent the temporal validity of the relations. Each row corresponds to a predicate characterising the relation (e.g., *commander*) to the specific event or entity (e.g., *Iraq War*).

and typically include events of major relevance from the personal perspective such as birth, education and career. Figure 6.1 illustrates a biography timeline for Barack Obama, which includes places where Barack Obama lived (first Chicago and then the White House), important events he was involved in (e.g., the Iraq War) and the major political positions he held (e.g., the President of the United States). This timeline also indicates the temporal validity of these relations.

First, we present an approach for the generation of biography timelines from a temporal knowledge graph. To generate such timelines, we propose a distant supervision method, where we train the relevance model using external sources containing biographical and encyclopaedic texts. With that model, we extract the most relevant biographical data from the temporal knowledge graph concisely describing a person's life, while using features such as relation strength and event popularity information contained in *EventKG*, as well as predicate labels. The results of our user evaluation demonstrate that this approach is able to generate high-quality biography timelines while significantly outperforming a state-of-the-art baseline for timeline generation: our timelines were preferred over the baseline's timelines in approximately 68% of the cases.

The process of making knowledge graphs accessible does not stop with the creation of timelines: there is a need for creating interfaces which visualise timelines and makes them available to non-expert users. To this end, we introduce two applications based on timelines extracted from *EventKG*.

Contributions. Our contributions in this chapter are as follows:

1 We define the problem of biography timeline generation from a temporal knowledge graph and present our method based on distant supervision.

2 We demonstrate the effectiveness of the proposed timeline generation method in a user study.

3 We demonstrate the usefulness of timeline-based applications at the example of two interactive interfaces.

Outline. The remainder of this chapter is organised as follows: First, we provide specific background for the task of timeline generation in Section 6.2. Then, we demonstrate how to access knowledge graphs without the provision of any visual interfaces by means of two example SPARQL queries on *EventKG* (Section 6.3). In Section 6.4, we define the problem of biography timeline generation, followed by our approach to solve that problem in Section 6.5. The experimental setup and evaluation of the biography timelines generated with our approach using *EventKG* are provided in Section 6.6. Afterwards, we demonstrate two interfaces based on *EventKG*: *EventKG+BT* (Section 6.7) and *EventKG+TL* (Section 6.8). Finally, we provide a conclusion and discuss our findings in Section 6.9.

6.2 Specific Background

Existing work on timeline generation from knowledge graphs has mainly focused on the selection of relevant events or relations. The works of Althoff et al. [ADM+15] and Tuan et al. [TEPW11] come closest to our task definition. With the TimeMachine system [ADM+15], the authors create timelines for politicians, actors and athletes from the Freebase knowledge graph, adding visual and diversity constraints on the generated timelines. An example of a timeline generated by TimeMachine is shown in Figure 2.9. In [TEPW11], person timelines are generated by ranking relations extracted from Wikipedia and YAGO knowledge graphs. Similarly, in [TLR16], entity summarisation is created based on link counts, but without taking temporal data into account. In difference to our work, in both these approaches, the feature weights are handcrafted, and no machine learning is involved. [CRH17] and [LGA16] aim at generating biographies in a natural language, that means to generate textual summaries for people, by mapping facts from knowledge graphs to one-sentence biographies. Both works incorporate neural models to learn text, but the biographies are limited to a few facts such as birth dates and entity types. Similarly, Pantheon is a manually verified biography dataset, that is limited to non-temporal facts and features and does not provide relations between persons and events [YRH+16].

Other approaches generate timelines for different use cases, for example, to get an overview of news articles over a large time span [TAH15, SA00] or for depicting

singular events such as football matches in a very fine-grained manner [AS13]. For visualisation, there are approaches to transform relationship paths from knowledge graphs into sentences [ADM+15, VMdR17] and different interaction models that let a user explore the timeline [ADM+15, ZDFB12, SA00]. In this chapter, we focus on the generation of timelines containing relevant temporal relations and do not limit the approach by any visual constraints. This way, the models obtained by our methods can be used in a broader range of interfaces and application scenarios.

One important subtask of the timeline generation is to judge whether a temporal relation is relevant in a particular context. This task has been addressed by other works using classification and ranking approaches. For example, to rank news articles related to a query entity, Singh et al. [SNA16] employ a diversified ranking model based both on the aspect and temporal dimension. Approaches such as the one proposed by Setty et al. [SAMA17] impose methods to rank the importance of events, but without taking into account the specific timeline entity. In comparison to these approaches, the task addressed in our work is more specific, as it considers the relevance of individual temporal relations to a timeline entity.

Biography and Timeline Visualisation. Few systems exist that provide visualisations of biography timeline extracted from knowledge graphs: BiographySampo [HLT+19] provides Finnish textual biographies that a user can explore using network exploration and maps. The TimeMachine by Althoff et al. [ADM+15] gives a compact overview of only a few related entities but does not provide time intervals, or any further information.

6.3 Example Queries for *EventKG*

To get an impression of what it means to manually query a knowledge graph, and to get a first idea of creating a biography timeline directly from *EventKG*, we present two example SPARQL queries that illustrate the retrieval of particular event and entity characteristics.

6.3.1 Query 1: Provenance and Event Locations

The SPARQL query in Listing 6.1 uses the named graph notation to find the locations of the event "Second inauguration of Barack Obama" in any source. This is done using the `sem:hasPlace` predicate introduced in Section 3.5.1. Table 6.1 lists the query results from *EventKG* V1.1. While YAGO has the United States Capitol and Washington, D.C., as location, Wikidata has Washington D.C. only. There are no locations for this event found in any of the DBpedia language editions. After fusion, the union of potential locations (United States Capitol and Washington, D.C.) is reduced to the United States Capitol only, which is located in Washington, D.C[1].

[1]This information could be inferred using `so:containedInPlace` transitively.

Fused locations are placed within *EventKG*'s named graph.

Table 6.1. Locations of the first inauguration of Barack Obama in *EventKG*.

?location	?named_graph
dbr:United_States_Capitol	eventKG-g:event_kg
dbr:Washington,_D.C.	eventKG-g:wikidata
dbr:United_States_Capitol	eventKG-g:yago
dbr:Washington,_D.C.	eventKG-g:yago

6.3.2 Query 2: Important Events of an Entity

The second query shown in Listing 6.2 employs the relation strength information contained in *EventKG*. It returns a list of events connected to Barack Obama, sorted by the number of common mentions (`eventKG-s:mentions`) with Barack Obama in the English Wikipedia (`GRAPH eventKG-g:wikipedia_en`). Additionally, if there is an event start date available, this is returned as well, using the named *EventKG* graph to retrieve the fused date. The results from *EventKG* V1.1 in Table 6.2 reveal that the United States presidential election of 2008 is the event mentioned most often together with Barack Obama.

```
SELECT ?location ?named_graph

WHERE {
  ?event owl:sameAs dbr:First_inauguration_of_Barack_Obama .

  GRAPH ?named_graph {
    ?event sem:hasPlace ?loc
  } .

  GRAPH eventKG-g:dbpedia_en {
    ?loc owl:sameAs ?location .
  }
}
ORDER BY ?named_graph
```

Listing 6.1: SPARQL query for retrieving the locations of the first inauguration of Barack Obama using `sem:hasPlace`, together with their named graph for provenance information.

Table 6.2. Events that are most often mentioned together with Barack Obama.

?event	?cnt	?startDate
dbr:United_States_presidential_election,_2008	719	2008-11-04
dbr:United_States_presidential_election_in_New_Jersey,_2012	530	2012-11-06
dbr:United_States_presidential_election_in_New_Jersey,_2008	522	2008-11-04
⋮		
dbr:First_inauguration_of_Barack_Obama	68	2009-01-20
⋮		

```
SELECT ?event ?cnt ?startDate

WHERE {
  ?obama owl:sameAs dbr:Barack_Obama .
  ?relation rdf:subject ?obama .
  ?relation rdf:object ?eventEKG  .

  GRAPH eventKG-g:wikipedia_en {
    ?relation eventKG-s:mentions ?cnt .
  }

  ?eventEKG rdf:type sem:Event .

  GRAPH eventKG-g:dbpedia_en {
    ?eventEKG owl:sameAs ?event
  } .

  OPTIONAL {
    GRAPH eventKG-g:event_kg {
      ?eventEKG sem:hasBeginTimeStamp ?startDate
    }
  } .
}
ORDER BY DESC(?cnt)
```

Listing 6.2: SPARQL query for retrieving the events that are most often mentioned together with Barack Obama. Instances of `eventKG-s:Relation` are searched who are connected to Barack Obama as their subject and an instance of `sem:Event` as their object.

From the results presented in Table 6.2, a biography timeline could be created by ordering the results with respect to the `?startDate` column. However, this procedure

does not only require SPARQL expertise but it also does not limit the timeline to specifically relevant temporal relations, and Table 6.2 only covers temporal relations of a specific type.

6.4 Problem Statement

Biography timeline generation facilitates the creation of chronological timelines of relevant relations and events in the life of a person.

Given a temporal knowledge graph $G_T = (E_t, R_t)$ as in Definition 3.1, we denote the temporal entity of user interest $e \in E_t$ for which the biography timeline is generated as a *timeline entity*.

A biographiy timeline is a chronologically ordered list of temporal relations involving the timeline entity and relevant to that entity's biography.

Definition 6.1. *A* ***biography timeline*** $TL(e, bio) = (r_1, \ldots, r_n)$ *of a timeline entity* e *is a chronologically ordered list of timeline entries (i.e., temporal relations involving* e*), where each timeline entry* r_i *is relevant to the entity biography bio.*

In this chapter, we assume a binary notion of relevance, i.e., $\forall r_i \in TL(e, bio) : relevance(e, r_i, bio) = 1$.

The list of timeline entries in $TL(e, bio)$ is ordered chronologically by their start time: $\forall r_i, r_j \in TL(e, bio) : i \leq j \Leftrightarrow r_{i_{start}} \leq r_{j_{start}}$.

An entity connected to e via a timeline entry r_i is denoted as a *connected entity* in the following.

6.5 Biography Timeline Generation: Approach

In this section, we show how *EventKG* can be applied as a temporal knowledge graph for the task of biography timelines generation.

First, we present our approach based on distant supervision in Section 6.5.1. The features used in the relevance model are introduced in Section 6.5.2. Subsequently, we describe the benchmarks involved in our process to generate biography timelines in Section 6.5.3 and discuss how the model is used to generate them in Section 6.5.4. Finally, we illustrate these steps on our running example of Barack Obama's timeline in Section 6.5.5.

6.5.1 Approach

Given a timeline entity e for which we need to generate a biography timeline, the number of *candidate timeline entries* (i.e., temporal relations involving e) is potentially

very high, especially for popular entities and a large-scale temporal knowledge graph. In fact, for our set of famous persons described later in Section 6.6.1, *EventKG* contains 272.75 temporal relations per person entity on average. In order to determine the relevance of a temporal relation to the timeline entity, we propose a classification approach using distant supervision. The key idea of our approach is to learn a relevance model for temporal relations using occurrences of these relations extracted from *biographical sources*. Examples of such biographical sources include collections of biographical or encyclopedic articles. We adopt a distant supervision approach, where we assume that a particular temporal relation r is relevant for the entity's biography if this relation occurs in a known biographical source. Figure 6.2 gives an example of this approach.

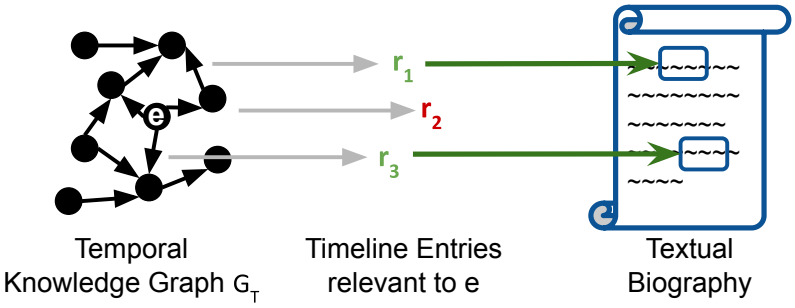


Figure 6.2. Distant supervision for relevance judgement of timeline entries: First, timeline entries relevant to the timeline entity e are extracted from the temporal knowledge graph. Then, their relevance is judged with respect to a biographical source: Here, r_1 and r_2 could be mapped to parts of the textual biography, and are thus marked ad relevant.

An overview of the training phase and the timeline generation is depicted in Figure 6.3, which illustrates the role of the temporal knowledge graph, the biographical and reference sources and the benchmark. Initially, we use the temporal knowledge graph and a biographical source to create a benchmark that provides relevance judgements for candidate timeline entries. We train the prediction model with features extracted for each candidate timeline entry. This includes entity type and interlinking information included in the named graphs corresponding to the reference sources of *EventKG*. To generate a timeline for a timeline entity e, we collect its candidate timeline entries R_e from G_T and identify the relevant entries using the trained model.

6.5.2 Relevance Model

In our approach, we train a classification model that identifies the relevance of a candidate timeline entry towards a biography of the timeline entity e. The candidate timeline entry is a temporal relation involving e and obtained from a knowledge graph. To train such classification models, we adopt a range of features in several categories

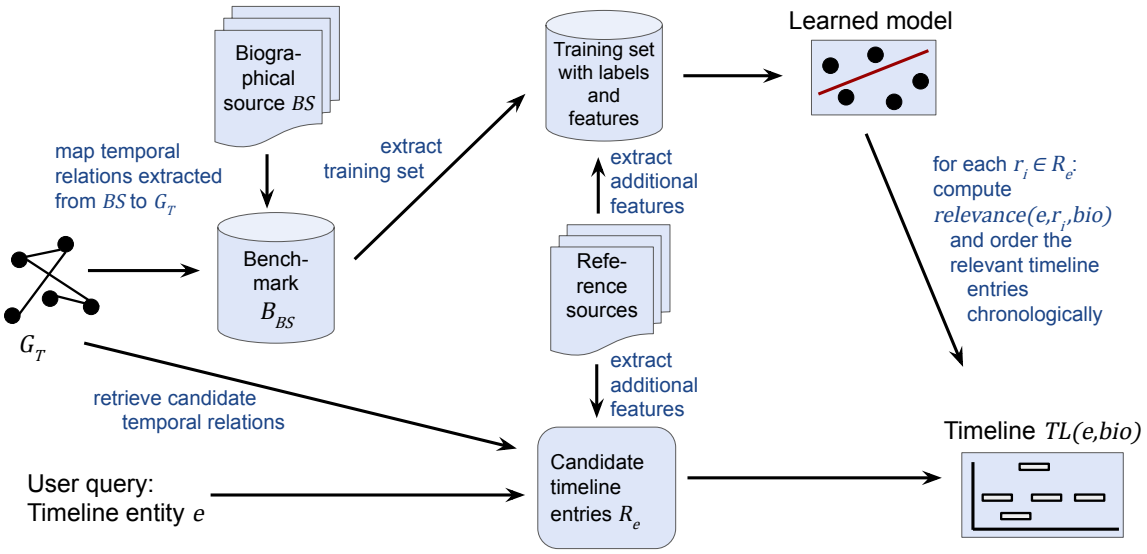


Figure 6.3. Creating a timeline for a timeline entity e, after training a model from a biographical source to predict the relevance of temporal relations in the temporal knowledge graph for biography timelines.

reflecting the characteristics of the timeline entity, the entity connected to it via a temporal relation, the temporal relation and time information. In total, we consider four language-independent numerical features, six language-dependent features, as well as a number of binary features representing frequent entity types and properties in *EventKG*.

We illustrate the features described in the following at the example of the candidate timeline entry representing Barack Obama's participation in his second inauguration (see Figure 3.2) in Table 6.3.

Timeline Entity Features

The timeline entity features (TEF) reflect specific characteristics of the timeline entity e. These features address the intuition that the relevance of the particular temporal relation r for a given timeline entity e depends on the specific characteristics of e. For example, winning an award may be more important for athletes or actors than for politicians. Based on this intuition, we introduce the timeline entity features:

TEF-C Timeline entity characteristics: A set of binary features denoting if the entity is an instance of the specific type (e.g., a politician or an actor).

Connected Entity Features

The connected entity features (CEF) take into account the characteristics of the connected entity e'. In particular, we consider indications of the importance and popularity of e' in the context of the reference collections by using mention counts, similar to Thalhammer et al. [TLR16]. We consider different representations of the mention counts of e':

CEF-M Connected entity mentions: The set of features, each reflecting the absolute number of mentions of the connected entity e' in a reference collection.

CEF-MR Connected entity mentions rank: For each reference collection, we rank the entities connected to the timeline entity e by the number of their mentions. This feature represents the rank of the specific connected entity, where the rank of 1 is assigned to the entity with the highest number of mentions.

CEF-MRR Connected entity mentions relative rank: We normalise the CEF-MR rank by the maximal rank.

CEF-E Connected entity represents a real-world event: A binary feature denoting whether the connected entity is an event (i.e., $e' \in \mathcal{V}$).

Features of Temporal Relations

The features of temporal relations (TRF) reflect the semantics of the temporal relation between the timeline entity and the connected entity. Furthermore, we consider features related to the importance and popularity of entity relations.

TRF-PI Property identifier: Temporal relations possess property identifiers r_{uri} that express the semantics of the relation (e.g., `dbo:spouse`). Each property identifier is modelled as a binary feature.

TRF-M Relation mentions: The number of co-mentions of both entities involved in the temporal relation in a reference collection (independent of relation semantics).

TRF-MR Relation mentions rank: We rank the connected entities according to the number of their co-mentions with the timeline entity in a reference collection. This feature represents the rank of the specific connected entity involved in the relation.

TRF-MRR Relation mentions relative rank: We normalise the TRF-MR rank by its maximal rank.

Temporal Features

The temporal features (TF) reflect the relevance of the temporal relations based on the time information. This includes the temporal differences in the existence time of the entities or happening times of the events involved in the relation. For example, Barack Obama gave a speech related to World War II – a historical event finished before Obama's birth date in 1961. Here, the temporal difference in the existence times of both entities can be an indication of the low relevance of this speech for Obama's biography. Therefore, we attempt to learn to discard the temporal relations involving events that happened too early for the entity timeline. This had also been observed by Althoff et al. [ADM+15], who implemented a rule to discard such relations. In addition to that, our temporal features could help to learn whether some events may be more relevant at specific stages of the entity's life or existence. Furthermore, temporal features include the provenance of the temporal information by denoting whether a relation was induced from an indirect temporal relation (as described in Section 3.5.2) or not.

To capture this intuition, we introduce the following temporal features:

TF-TDS Temporal distance (start): The temporal distance between the beginning of the existence time of the timeline entity and the start of the relation validity time $e_{start} - r_{start}$.

TF-TDE Temporal distance (end): The same feature as TF-TDS, but using the entity existence end time $e_{end} - r_{start}$.

TF-TP Time provenance: This categorical feature specifies the provenance of the relation validity time. If the relation has initially been a temporal relation, the feature value is set to 3. If the temporal validity was induced from an event happening time ($e_j \in \mathcal{V}$), then the feature value is set to 2; 1 otherwise ($e_j \in \mathcal{E}'$).

6.5.3 Benchmarks for Distant Supervision

To facilitate supervised model training, we require a benchmark that provides relevance judgements for temporal relations. These judgements can be obtained from the specific biographical source.

Definition 6.2. *A **benchmark** B is a mapping of the form: relevance$(e_i, r_j, bio) \mapsto J, J \in \{0, 1\}$, where e_i is a temporal entity, r_j is a temporal relation involving e_i and J is a relevance judgement.*

Given the large number of entities and temporal relations in the existing knowledge graphs, manual relevance judgements appear unfeasible. Therefore, we adopt an automatic approach to benchmark generation. We extract entities and temporal

Table 6.3. Selected feature values for the candidate timeline entry "Barack Obama, significant event, Second inauguration of Barack Obama" for the timeline entity "Barack Obama".

Feature	Feature Instance	Value	Note
TEF-C	Politician	1	Barack Obama is an instance of `dbo:Politician`.
	President	1	Barack Obama is an instance of `dbo:President`.
	Scientist	0	Barack Obama is not an instance of `dbo:Scientist`.
CEF-M	CEF-M_{EN}	84	The inauguration is linked 84 times in the English Wikipedia.
CEF-MR	CEF-MR_{EN}	361	Among all entities connected to Obama in the English Wikipedia, the inauguration is linked the 361st most times.
CEF-MRR	CEF-MR_{EN}	0.817	Among all entities connected to Obama in the English Wikipedia, there are 442 different CEF-MR_{EN} scores, such that inauguration's relative rank is $\frac{361}{442} \approx 0.817$.
CEF-E	CEF-E	1	The inauguration is an instance of `sem:Event`.
TRF-PI	`wd:significantEvent`	1	Obama is connected to the inauguration through Wikidata's "significant event" property.
	`wd:spouse`	0	Barack Obama is not connected to the inauguration through Wikidata's "spouse" property.
TRF-M	TRF-M_{PT}	4	In the Portuguese Wikipedia, there are 4 sentences mentioning both Barack Obama and the inauguration.
TRF-MR	TRF-MR_{PT}	18	Among all co-mentions of Barack Obama and an event, the co-mention with the inauguration is the 18th most frequent one the Portuguese Wikipedia.
TRF-M	TRF-M_{ALL}	36	In all the five involved Wikipedia language editions together, there are 36 sentences mentioning both Obama and the inauguration.
TRF-MR	TRF-MR_{ALL}	39	Among all co-mentions of Barack Obama and an event, the co-mention with the inauguration is the 39th most frequent one in all the five involved Wikipedias together.
TF-TDS	TF-TDS	18798	The inauguration started 18798 days (51 years) after Barack Obama's birth.
TF-TDE	TF-TDE	18798	The inauguration ended 18798 days (51 years) after Barack Obama's birth.
TF-TP	TF-TP	2	The validity time assigned to this temporal relation is induced from the happening time of an event instance.

relations contained in the biographical sources and map them to the temporal relations in G_T using an automatic procedure involving source-specific heuristics (described later in Section 6.6.1). Temporal relations extracted from the biographical sources are considered relevant.

Although the resulting benchmarks can potentially contain noisy relevance judgements due to the automatic extraction and mapping methods applied, our experimental results demonstrate that these benchmarks, used as a training set in a distant supervision method, facilitate the generation of high-quality timelines.

The benchmarks created in this work are publicly available online[2].

6.5.4 Model Training and Timeline Generation

We address the relevance estimation for a timeline relation r with respect to the timeline entity e as a classification problem. For each biographical source BS, we build a classification model using the features presented in Section 6.5.2 and a binary classifier.

Note that a classification model is chosen over a ranking-based approach because of two reasons: First, the timeline entries are ordered chronologically and not by their importance. Therefore, for the purpose of timeline generation, we can assume that each timeline entry is equally relevant. Second, if a ranked list of timeline entries would be provided, a cut-off threshold value would still be required to decide which of the entries are to be shown.

To facilitate efficient training, we limit the number of instances of the TEF-C and TRF-PI features considered. In particular, the 50% most frequent types in the training set are added as a TEF-C feature. Furthermore, only properties that occur in at least 25% of the relations in the training set are added as a TRF-PI feature.

Our benchmark is equally divided into a training and a test set of person entities so that the relevance judgements are obtained from the training set. We adopt a binary notion of relevance. The datasets used as biographical sources to build the classification models are presented in Section 6.6.1.

We use the resulting classification model to build a timeline $TL(e, bio)$. Each candidate timeline entry (i.e., a temporal relation involving the timeline entity e in G_T) is classified using the classification models learned from a biographical source. The classification function $relevance(e, r, bio)$ uses this model to classify the temporal relations of the timeline entity e as either 0 (non-relevant) or 1 (relevant). As illustrated in Figure 6.3, the timeline is generated by ordering the timeline entries classified as relevant by their start time.

6.5.5 Running Example: Barack Obama

Following Section 3.5.2, we extract temporal relations from *EventKG* as the union of the following three relation types:

1. Relations where the object is a temporal literal. Example:

[2] http://eventkg.l3s.uni-hannover.de/timelines.html

- *Barack Obama*, `born`, *4 Aug 1961*

2. Relations that are directly assigned a validity time span. Example:
   - *Barack Obama*, `marriedTo`, *Michelle Obama* [3 Oct 1992 –]
3. Indirect temporal relations where the validity time is identified using the object's happening or existence time. Example:
   - *Barack Obama*, *child*, *Malia Ann Obama* [4 Jul 1998 –]

As discussed in Section 3.5.4, *EventKG* contains many relations involving Barack Obama. In order to create a timeline of his life, we collect all relations with Obama as a subject or an object, together with their temporal validity. Another example is the temporal relation about Obama's first inauguration shown at the end of Section 3.5.4.

As this procedure leads to more than 2,500 candidate timeline entries for Barack Obama, we now need to apply the previously trained model to determine the timeline entries relevant for a biography. To this end, we train the classifier that predicts whether a candidate timeline entry is relevant given a biographical source, i.e., whether it is probable to be part of entity biography in such source. All candidate timeline entries that are classified as relevant by this model are inserted into the timeline in chronological order.

Figure 6.1 provides a visual representation of Obama's timeline obtained using a model trained on a Wikipedia abstracts dataset (BS-ENC) described later in Section 6.6.

6.6 Evaluation

In this section, we first describe the biographical sources and the set of timeline entities used to create our biography timeline benchmark used to train the classification models (Section 6.6.1) and to run our experiments described in Section 6.6.2. Then, we evaluate our approach against a baseline (Sections 6.6.3 and 6.6.4).

6.6.1 Benchmark: Entities and Biographical Sources

We collect a dataset $\mathcal{P}$ that contains 2,760 timeline entities of the type *Person*, including its subtypes like politicians, actors, musicians and athletes. This set of 2,760 entities contains all persons that are included in *EventKG* and described in each biographical source described below. Consequently, the training and the test set consist of 1,380 person entities each, after random division.

To train the relevance models for the biography timeline generation, we consider the following biographical sources:

- BS-BIO: Biographical articles;
- BS-ENC: Encyclopedic articles.

Biographical articles (BS-BIO):

Biographies of important entities (e.g., famous people) are available in the form of textual descriptions from dedicated Web sources. We collect data from two publicly accessible biographical web sources (Thefamouspeople.com[3] and Biography.com[4]). After collecting the biographical texts from both websites, they are pre-processed as follows: 1) The texts are split into sentences using the Stanford Tokenizer [MSB+14]. 2) Time expressions are collected from each sentence using HeidelTime [SG10]. 3) Entity mentions are identified using DBpedia Spotlight [MJGSB11]. Table 6.4 illustrates example annotations in the *BS-BIO* and *BS-ENC* datasets extracted for the entity Barack Obama, including his birth, education and political activities. In order to map the extracted information to the temporal relations in the temporal knowledge graph (as illustrated in Figure 6.2), we use the following rule-based approach:

Table 6.4. Example data extracted from the biographical sources for Barack Obama.

	BS-BIO	**BS-ENC**
Source	biography.com, thefamouspeople.com	$\text{Wikipedia}_{\text{EN}}$ abstracts
Example Data	1961-8-4, {Honolulu} 1979, {Punahou School, Basketball} 2000, {Democratic Party, Bobby Rush} 2010-8, {War in Afghanistan, Iraq}	1961, {Honolulu} 2013, {US presidential election 2012, Mitt Romney, Second inauguration of Barack Obama} 2009, {Nobel Peace Prize}

An annotated sentence in the biographical article is mapped to the temporal relation in G_T if they both happened on exactly the same date, or if they share both entities and time. A special case is given if one of the linked entities is an event in $\mathcal{V}$. In that case, the temporal overlap is not required, as events are typically inherently connected to a validity time span. The mapped temporal relations from G_T are added to the B_{BS-BIO} benchmark.

Encyclopedic articles (BS-ENC):

Wikipedia is a rich source of encyclopedic information. Wikipedia articles usually provide an abstract – a brief overview of the specific entity (e.g., a person's life) that typically contains important biographical sentences [CRH17, LGA16]. From these

[3] www.thefamouspeople.com
[4] www.biography.com

Table 6.5. Statistics of the dataset $\mathcal{P}$ involving 2,760 entities of type person.

	thefamouspeople.com	biography.com	Wikipedia Abstracts
Time expressions	50,919	41,318	18,099
Entity links	107,126	92,149	32,516

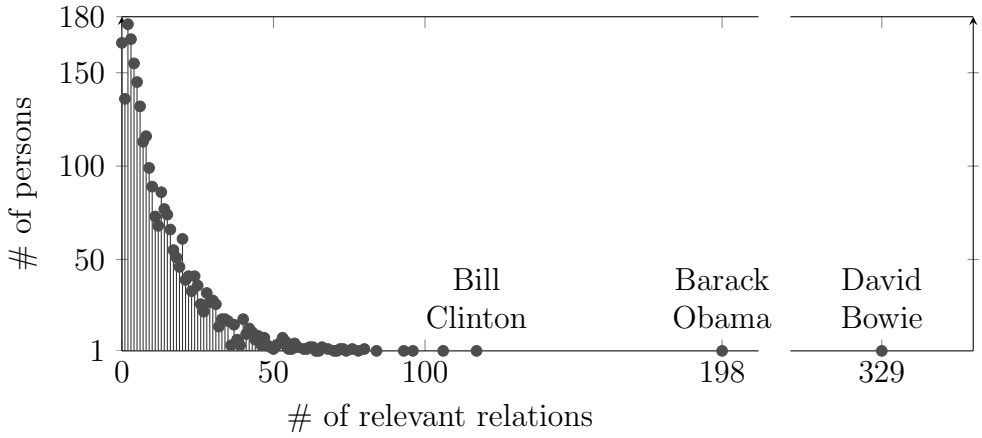


Figure 6.4. The number of person entities with the given number of relevant relations in the BS-BIO benchmark. The top-3 entities with the highest number of relevant relations are marked.

abstracts, we extract all the event mentions, i.e., links to the event articles, as these represent significant events in the entity's life. For example, Table 6.4 shows selected events for the entity Barack Obama based on BS-ENC. In contrast to the annotations in B_{BS-BIO}, these events are more focused on political happenings with major public impact. The benchmark B_{BS-ENC} includes all relations of the specific entity to the events linked from the abstract of the Wikipedia article representing this entity.

Statistics of the entity-related information for the entities contained in the dataset $\mathcal{P}$ in the biographical sources, including in particular the number of relevant entity links and time expressions is provided in Table 6.5.

Figure 6.4 illustrates the distribution of the number of relevant relations per person in the BS-BIO benchmark. Except for very few popular entities such as David Bowie and Barack Obama, the number of relevant relations is typically below 100, with an average of 13.64.

We generate a benchmark B_{BS} for each biographical source BS considered in this work. The statistics regarding these benchmarks are presented in Table 6.6.

Table 6.7 provides the percentage of person types in the benchmarks. Actors and musical artists are the most frequent person types in both the training and test set.

Table 6.6. Benchmark statistics: the number of entities and relevant temporal relations.

	# Persons	#Relevant Temporal Relations	Average # Temporal Relations per Entity
B_{BS-BIO}	2,760	37,638	13.64
B_{BS-ENC}	2,760	33,106	12.00

Table 6.7. Percentage of top-5 entity types in the training and test set.

	Training	Test
Actor	27.73%	28.57%
Musical Artist	13.32%	16.17%
Athlete	10.50%	6.16%
Politician	10.35%	10.44%
Writer	6.95%	11.31%

6.6.2 Classifier Setup and Timeline Statistics

As our binary classifier, we adopted a Support Vector Machine (SVM) due to its good generalisation ability, in particular when applied to smaller datasets. We trained this classifier on the training dataset containing 1,380 person entities, with input data normalisation, an increased weight of 3.0 for predicting relevant instances and a linear kernel, using Weka's LibSVM implementation [WFHP16]. From the training data, a balanced set of relevant and irrelevant instances is given to the SVM.

As described in Section 6.5.4, the timelines are generated by ordering the timeline entries classified as relevant chronologically by their start time. On average, each biography timeline of the person entities in the test set contains 8.54 entries after training the classifier on B_{BS-BIO} (B_{BS-ENC}: 7.81). Figure 6.5 illustrates the number of timelines generated for the $BS-BIO$ with the specific number of entries.

6.6.3 The TM Baseline Algorithm

We compare our proposed approach with the state-of-the-art Time Machine (TM) approach for timeline generation proposed by Althoff et al. [ADM+15] and shown in Figure 2.9. The TM approach creates events from the entity-entity relations in a knowledge graph, where one entity possesses a property with a time value. Resulting events are filtered using frequency and existence time heuristics; then, a greedy algorithm selects the events that maximise a relevance score. To facilitate a fair comparison, we perform the following adjustments to implement the TM baseline:

- The TM approach in [ADM+15] was initially proposed for entity-centric knowledge graphs such as Freebase. Therefore, events in the TM terminology mean

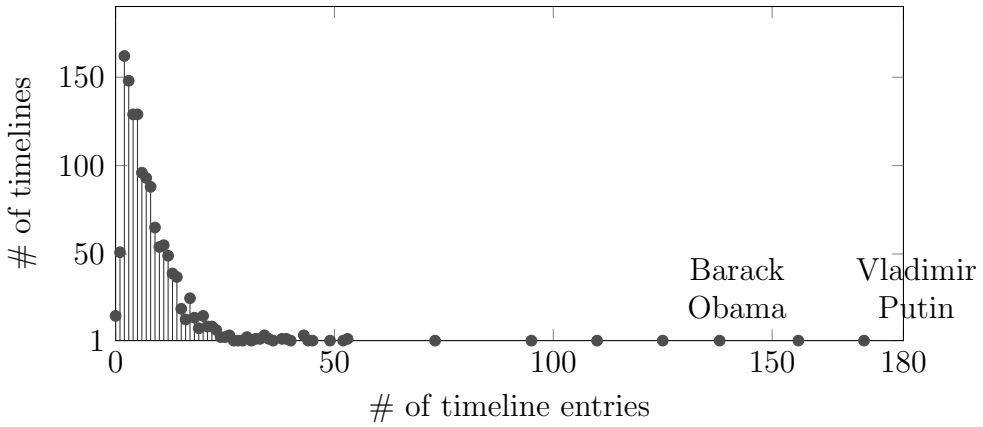


Figure 6.5. The number of timelines with the specific number of entries generated for the $BS - BIO$ test set.

link structures in an entity-centric knowledge graph that vary with respect to their complexity. In *EventKG*, the events are connected to the entities directly via temporal relations. To facilitate the comparison, we adopt the TM baseline such that so-called "simple events" in the TM-terminology are generated. Such "simple events" in TM directly correspond to the temporal relations in *EventKG*.

- In the original TM approach, the maximum number of temporal relations on the timeline is restricted due to the visualisation constraints; i.e., these relations are ranked by their relevance and retrieved until the visualisation constraint is met. Our goal is to provide all relevant relations, such that we do not enforce any visualisation-based constraints on the number of relations. To facilitate comparison, we retrieve an equal number of relations from the baseline and our approach.

- TM was initially evaluated on the Freebase dataset, and the relevance scores were computed using a search engine query log and a textual corpus. We apply all methods on the *EventKG* data; we use the same reference sources (i.e., Wikipedia articles) to estimate the parameters related to the global importance of entities, their occurrences and temporal relations for all baselines and approaches evaluated in this chapter.

6.6.4 Evaluation of the Timeline Generation

The goals of the evaluation of the timeline generation are to assess the effectiveness of the proposed method for timeline generation and the role of the reference and biographical sources.

In particular, we assess:

G1 Quality of the generated timelines in comparison to the baseline (in a user evaluation).

G2 Impact of the individual features on the timeline generation (using correlation measures).

G3 Relevance of the timeline entries with respect to the biographical source (by measuring the performance of the classification model).

G4 Coverage of the timeline entries with respect to the reference sources (by measuring the mean coverage of the temporal relations in the reference sources).

Timeline Quality Evaluation

In order to evaluate the timeline quality, we performed a user evaluation. We generated timelines for 60 popular entities of the types actors, athletes, musical artists, politicians and writers for both biographical sources BS-BIO and BS-ENC. These entities were selected from the persons in the test set described in Section 6.6.1 based on their popularity (measured as the link count of the corresponding Wikipedia article).

In each task, the user was presented with: (i) a task description, (ii) a timeline entity including its label and a Wikipedia link, and (iii) a pair of timelines. One timeline in the pair was generated by the specific configuration of our approach, the other timeline was generated by the TM baseline described in Section 6.6.3. Both timelines were visualised as illustrated in Figure 6.1. Each timeline contained all entries generated by the corresponding generation method. The user could scroll and zoom within each individual timeline. In the user interface, both timelines were presented simultaneously, one above the other, in random order. We asked the users to vote for their preferred timeline in the pair. We provided four options: two options to vote for one of the timelines, a neutral option indicating no preference for a specific timeline, and a "don't know" option. We encouraged the users to research the timeline entity (e.g., using Wikipedia) before evaluating the timeline pair, if necessary.

Each pair of timelines was rated by three or four users each. Then, majority voting was applied. In total, 11 users (graduate Computer Science students) participated in the user evaluation. A user evaluated 42 timeline pairs on average. On average, the users took 69 seconds to decide between two timelines.

We compute the rater preference $RPref$ score adopted from [ADM+15] as the fraction of votes for the particular method, based on the annotation that is most frequent among the three users per timeline entity. The results of the user evaluation are presented in Table 6.8. The timelines generated by our approach with both biographical sources (BS-BIO and BS-ENC) were preferred over the baseline by the users most of the time, for all entity types. For example, all of the 16 timelines for politicians generated by our approach with BS-ENC were preferred over the TM

Table 6.8. RPRef scores from user ratings for different timeline configurations and entity types. As users could also give a neutral rating or skip a rating, the RPRef scores do not necessarily sum up to 100%.

Biographical Source	**BS-BIO**		**BS-ENC**	
Method	**BS-BIO**	**TM baseline**	**BS-ENC**	**TM baseline**
Actor	81.82%	9.09%	72.73%	9.09%
Athlete	75.00%	8.33%	58.33%	25.00%
Musical Artist	70.00%	0.00%	50.00%	30.00%
Politician	53.33%	13.33%	100.00%	0.00%
Writer	61.54%	30.77%	53.85%	25%
Total	**67.21**%	13.11%	**69.35**%	14.52%

timelines. In total the timelines from BS-BIO were preferred in 67.21% of the cases, and the BS-ENC timelines were preferred in 69.35% of the cases.

For BS-BIO, the mean number of ratings favouring our timeline is 1.50 (BS-ENC: 1.58) with a standard deviation of 0.72 (BS-ENC: 0.97), for the TM baseline, the mean is 0.40 (BS-ENC: 0.59) with a standard deviation of 0.67 (BS-ENC: 0.74). The results of the paired t-test confirm the statistical significance of this result for the confidence level of 99%.

Feature Impact

In total, 411 features are utilised by the model during the timeline generation. In order to better understand the impact of the individual features on the classification task, we compute the correlation between the features and the benchmark judgements using the Pearson Correlation Coefficient ($PCC \in [-1, 1]$, with $PCC = 0$ corresponding to no linear relationship), shown in Table 6.9.

For both biographical sources, the highest PCC is achieved for the property `born` ($PCC = 0.39$ for BS-ENC, $PCC = 0.25$ for BS-BIO). The `died` property and the time provenance feature TRF-TP are of similar relevance in both biographical sources, followed by the features related to relation mentions. In contrast, properties like `cover artist` and `draft team` do not correlate with the relation importance. One interesting difference between the biographical sources is the property `spouse` that is highly relevant in the biographical source BS-BIO but is ranked lower in BS-ENC. Such personal happenings are often not included in Wikipedia's encyclopedic abstracts.

Relevance of the Timeline Entries

We evaluated the performance of the classification models for predicting the relevance of the individual temporal relations with respect to the benchmarks presented in

Table 6.9. PCC correlation coefficient between top-5 features and the benchmark judgments, sorted by the absolute PCC values (first column: Rank).

	BS-BIO		BS-ENC	
	Feature	**PCC**	**Feature**	**PCC**
1	TRF-PI: *born*	0.25	TRF-PI: *born*	0.39
2	TF-TP: Time provenance	0.21	TRF-PI: *died*	0.27
3	TRF-PI: *died*	0.19	TF-TP: Time provenance	0.23
4	TRF-MR: Relation mentions rank, EN	-0.19	TRF-MR: Relation mentions rank, EN	-0.19
5	TRF-MR: Relation mentions rank, all	-0.18	TRF-MR: Relation mentions rank, all	-0.18
	...			
10	TRF-PI: *spouse*	0.13	TRF-MR: Relation mentions rank, RU	-0.14
	...			
65	TRF-PI: *director*	0.03	TRF-PI: *spouse*	0.03
	...			
410	TRF-PI: *cover artist*	0.00	TRF-PI: *military rank*	0.00
411	TRF-PI: *illustrator*	0.00	TRF-PI: *draft team*	0.00

Section 6.6.1. The results of this automated evaluation using 10-fold cross-validation are presented in Table 6.10. In general, our models learned from the training set are generalisable to the test set, reaching F-measure values of 0.827 in the case of BS-ENC and 0.738 for BS-BIO. Across the biographical sources, the usage of all features combined leads to the best precision and recall scores. The removal of features leads to a decrease in performance: leaving out property labels or the features based on mentions leads to the biggest performance decrease.

Coverage of the Reference Sources

To demonstrate the gain of integrating data from multiple reference sources into *EventKG*, we assess the coverage of temporal relations in the biographical sources. That means, for each person in our benchmark, we compute the percentage of benchmark relations that are found in the temporal relations of a reference source. Table 6.11 shows the results, measured by mean coverage per person entity. For example, 27.45% of the relations extracted from BS-ENC can be mapped to a temporal relation in Wikidata. Additionally, we compute the coverage for *extended* reference sources, i.e., we still only consider relations from the specific source, but use the fused information about temporal entities (i.e., existence and happening times) from *EventKG*.

The results show that there is a higher coverage for BS-ENC than for BS-BIO across all reference sources. This can be explained by the fact that the texts from BS-BIO are longer and fewer event links are provided: not only does the BS-BIO benchmark rely on named entity recognition, as this source does not contain any links, but events are also harder to recognise as they can be described in several ways (e.g., "first inauguration of Barack Obama" and "Barack Obama was sworn in as the president on January 20, 2009"). In general, YAGO and Wikidata clearly outperform

Table 6.10. Weighted precision and recall scores for both classes (relevant and irrelevant) for predicting the benchmark labels of the temporal relations using a 10-fold cross-validation. Additionally, the F-measure as harmonic mean of precision and recall is reported. † All language-dependent features except for EN are omitted.

Features	Omitted Features	BS-BIO			BS-ENC		
		Precision	Recall	F-Measure	Precision	Recall	F-Measure
all features	/	0.796	0.749	**0.738**	0.848	0.829	**0.827**
no property labels	TRF-PI	0.753	0.691	0.671	0.822	0.802	0.799
no mentions	TRF-RM	0.769	0.700	0.679	0.802	0.734	0.719
no temporal features	TF-TP, TF-TDS, TF-TDE	0.795	0.747	0.736	0.847	0.829	0.827
English only	†	0.791	0.737	0.724	0.843	0.821	0.819

Wikipedia and DBpedia (as DBpedia does not contain statements with validity times). Through the integration and fusion in *EventKG*, the coverage increases to more than 50% in BS-ENC.

6.7 *EventKG+BT*: Biography Timeline Interface

In this section, we introduce *EventKG+BT*[5] – a system that enables exploration of the biography timelines. We demonstrate how the *EventKG+BT* system implements the distant supervision approach to biography timeline generation presented before and provides an interactive biography timeline. In this way, *EventKG+BT* can help to obtain a concise overview of a biography, alleviating the burden of time-consuming reading of long biographical or encyclopedic articles.

6.7.1 Biography Timelines

We assume a use case where the user task is to gain insights into the life of a person of interest, e.g., to get the first impression and a rough understanding of that person's role in history, the notable accomplishments, and to obtain a starting point for further in-depth research. To this extent, *EventKG+BT* shows a biography timeline to the user as the core of the visualisation. In total, *EventKG+BT* consists of several components that together enable interaction with the biography timeline. Figure 6.6 presents an example of the generated biography timeline for Barack Obama.[6]

[5] http://eventkg-biographies.l3s.uni-hannover.de

[6] The photo of Barack Obama is in the public domain (https://commons.wikimedia.org/wiki/File:President_Barack_Obama.jpg), the map is from OpenStreetMap, licensed under Attribution-ShareAlike 2.0 Generic (CC BY-SA 2.0, https://creativecommons.org/licenses/by-sa/2.0/).

Table 6.11. Mean coverage of the temporal relations in the benchmarks per reference source and biographical source.

	BS-BIO		BS-ENC	
	Mean Coverage (%)	**Mean Coverage (%) (extended)**	**Mean Coverage (%)**	**Mean Coverage (%) (extended)**
Wikidata	14.39	16.09	36.15	38.64
YAGO	11.96	12.34	37.90	38.40
Wikipedia$_{EN}$	0.51	14.56	0.80	23.65
Wikipedia$_{FR}$	0.34	11.04	0.61	18.96
Wikipedia$_{DE}$	0.16	0.86	0.40	16.66
Wikipedia$_{PT}$	0.00	8.61	0.16	15.73
Wikipedia$_{RU}$	0.22	8.68	0.43	15.41
Wikipedia	0.86	15.08	1.37	23.74
DBpedia$_{EN}$	5.05	9.27	27.94	34.97
DBpedia$_{FR}$	4.10	7.27	22.01	28.40
DBpedia$_{DE}$	4.48	6.41	25.69	28.90
DBpedia$_{PT}$	0.0	2.60	0.0	4.75
DBpedia$_{RU}$	0.0	1.48	0.0	2.64
DBpedia	5.73	14.53	30.02	45.10
EventKG	**23.29**	—	**55.09**	—

Wikipedia biography. On top, a brief textual biography and the person's Wikipedia link is shown next to the person's image.

Event map. An interactive map displays the locations of timeline entries and events in the person's life.

Biography timeline. The actual biography timeline is displayed in the centre. At first glance, the user can see the person's life span, as well as relevant phases in the person's life. Among other timeline entries, the example timeline indicates Obama's residences, as well as his term as US president. The user can interact with the timeline to obtain additional information.

Related people. Below the timeline, a list of people relevant to the selected person is shown to enable the exploration of further biography timelines.

Events. *EventKG+BT* also presents a chronological list of textual events in the person's life (e.g., "Senator Barack Obama officially announces his candidacy for president during a speech at the Old State Capitol in Springfield, Illinois.") that are queried from *EventKG*.

User Interaction and Data Export

The different components of *EventKG+BT* are connected and are highly interactive. For example, a click on a timeline entry leads to the selection of the associated location, event and people.

EventKG+BT does also offer an export option for the events and relations that underline the timeline generation, which provides access to the timeline facts in a JSON file. Moreover, the exported file contains all the temporal relations that were judged as non-relevant by our model. That way, we envision that *EventKG+BT* can facilitate further research on biography timeline generation from the knowledge graph.

6.7.2 Datasets and Implementation

EventKG+BT relies on models pre-trained on Wikipedia and biographical websites, temporal relations extracted on-the-fly from *EventKG* and additional information obtained from Wikipedia (the brief textual biography and image). The user can generate biography timelines for nearly 1.25 million persons. The pre-trained models were learnt on the benchmarks B_{BS-BIO} and B_{BS-ENC} consisting of 2,760 persons and more than 750 thousand biography entries introduced in Table 6.6.

EventKG+BT[7] is accessible as an HTML5 website implemented using the Java Spark web framework[8]. The biography timelines are visualised through the browser-based Javascript library vis.js[9], the maps are generated through the Leaflet Javascript library[10], and pop-overs showing detailed information are based on Bootstrap[11]. *EventKG* data is queried through its SPARQL endpoint[12], and Wikipedia information is retrieved via the MediaWiki action API[13]. To reduce the number of calls to the SPARQL endpoint, biography timelines are cached.

6.8 *EventKG+TL*: Event Timeline Interface

The amount of event-centric information regarding contemporary and historical events of global importance, such as the Brexit and the migration crisis in Europe, constantly grows on the Web, in Web archives, in the news as well as within emerging event-centric collections [GDR15] and knowledge graphs generated from these sources (e.g., [GD18a], [RvEV+16]). An important research area in this context is cross-cultural and cross-lingual event analytics (e.g., see [Rog13], [GDBR17] for case studies, and [GD17] for a cross-lingual user interface). These studies aim to analyze language-specific and community-specific representations and perceptions of historical and contemporary events, including their popularity and relations in a language context as well as to better understand the cross-lingual differences.

[7] http://eventkg-biographies.l3s.uni-hannover.de
[8] http://sparkjava.com/
[9] http://visjs.org/timeline_examples.html
[10] https://leafletjs.com
[11] https://getbootstrap.com/
[12] http://eventkg.l3s.uni-hannover.de/sparql.html
[13] https://www.mediawiki.org/wiki/API:Main_page

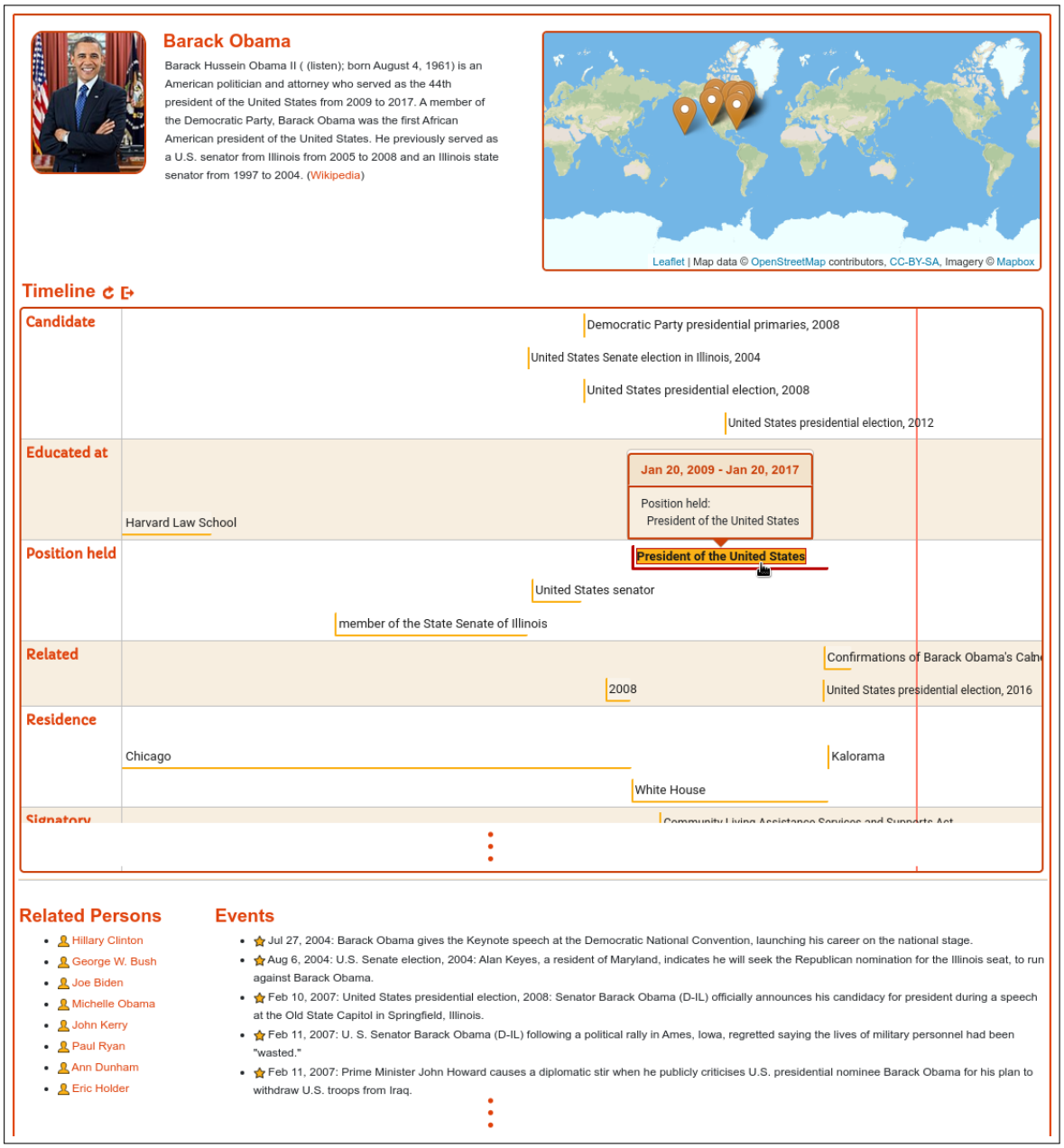


Figure 6.6. An excerpt of the biography timeline about Barack Obama, showing a short textual biography, a map, the generated biography timeline, related people and events. If possible, the timeline entries are grouped by property labels of the underlying temporal relations (e.g., "Candidate" and "Position held"). The "Events" section shows textual events related to Barack Obama, e.g., his announcement for presidential candidacy.

EventKG can facilitate a variety of studies and applications related to cross-cultural and cross-lingual event analytics. However, given a *query entity*, i.e., an entity or an event of user interest, *EventKG* can contain hundreds of related events along with their descriptions in several language contexts, which makes the provision of a comprehensive cross-lingual overview and a selection of relevant events for further detailed analysis challenging.

Timelines are an intuitive way to provide an overview of events related to a *query entity* over a certain period of time. However, existing timelines do not explicitly support a cross-lingual comparison of language-specific event representations, including their popularity and relation to the *query entity* in different language contexts.

EventKG+TL presented in this section is a timeline generator that creates cross-lingual timelines for a *query entity*, while relying on *EventKG* to provide language-specific information with respect to the event popularity and the relation strength between the events and the *query entity*. To this extent, *EventKG+TL* conducts a language-specific event ranking and complements this ranking with a cross-lingual visual representation. The timelines generated by *EventKG+TL* facilitate efficient identification of relevant events based on their language-specific popularity, relation strength and the cross-lingual differences.

6.8.1 Scenarios & Timelines

A *multilingual event-centric temporal knowledge graph* $G_L = (L, E, R)$ is a labelled directed multigraph, where L is a set of language contexts, E is a set of nodes (i.e., events or entities), and R is a multiset of directed edges (i.e., relations).

Given a *query entity* $q \in E$, the timelines generated by *EventKG+TL* can assist users in answering questions such as:

Q_1: *What are the most popular events related to q?*

Q_2: *Which events are the most closely related to q?*

Q_3: *Which of the most popular events are the most closely related to q?*

Q_4: *How does the popularity of the identified events and the strength of their relations to the query entity q differ across the language contexts?*

The provision of *EventKG+TL* facilitates users to answer these questions with respect to a particular language context $l \in L$ and enables a visual cross-lingual comparison. To answer these questions, the user of *EventKG+TL* can issue a *timeline query* that includes the following parameters:

- a *query entity* $q \in E$;
- a set of the language contexts of user interest $L' \subseteq L$;
- the maximum number k of the events to be selected per language context;

- the ranking criterion rc_i to identify the top-k most relevant events among all events $E' \subset E$ related to q in G_L according to the questions $Q_1 - Q_3$.

The ranking criteria include:

rc_1: *popularity*(e, l) is the popularity of an event $e \in E'$ in $l \in L'$;

rc_2: *relation strength*(q, e, l) is the relation strength between the *query entity* q and an event $e \in E'$ in a language context $l \in L'$; and

rc_3: *combined*(q, e, l) is a combination of the event popularity of $e \in E'$ and the relation strength between e and the *query entity* q in $l \in L'$.

The timelines generated by *EventKG+TL* complement the language-specific event ranking with a cross-lingual visual representation to address the question Q_4. To this extent, *EventKG+TL* utilises labelled pie charts located on a timeline, where each pie chart represents an individual event. The size of the pie chart corresponds to an overall (i.e., language independent) relevance of the event according to the ranking criterion rc_i. Each slice of the pie chart represents a language context. The area of each slice is proportional to the contribution of the corresponding language context to the ranking criterion rc_i.

Figure 6.7 exemplifies a timeline of the 2012 presidential elections in the United States. In the shown excerpt, we can observe that the most important event according to rc_3 are the *"2012 Summer Olympics"*, which is the most popular event in all considered language contexts[14], followed by the *"2012 Republican Party presidential primaries"*. Some of the events are more important in the specific language contexts, e.g., *"Death of Osama bin Laden"* in the German and *"Occupy Wall Street"* in the Russian context.

6.8.2 Timeline Generation

The Knowledge Graph

To answer a *timeline query*, *EventKG+TL* utilises *EventKG*. One of the key features of *EventKG* is the provision of event-centric information for historical and contemporary events, including their interlinking in the language-specific contexts to facilitate an assessment of relation strength and event popularity (see Section 3.5). The information on language-specific interlinking provided by *EventKG* is based on the corresponding Wikipedia language editions.

[14]The Olympics are connected to the elections through the hashtag #romneyshambles (https://www.newyorker.com/news/lauren-collins/romneyshambles-part-ii).

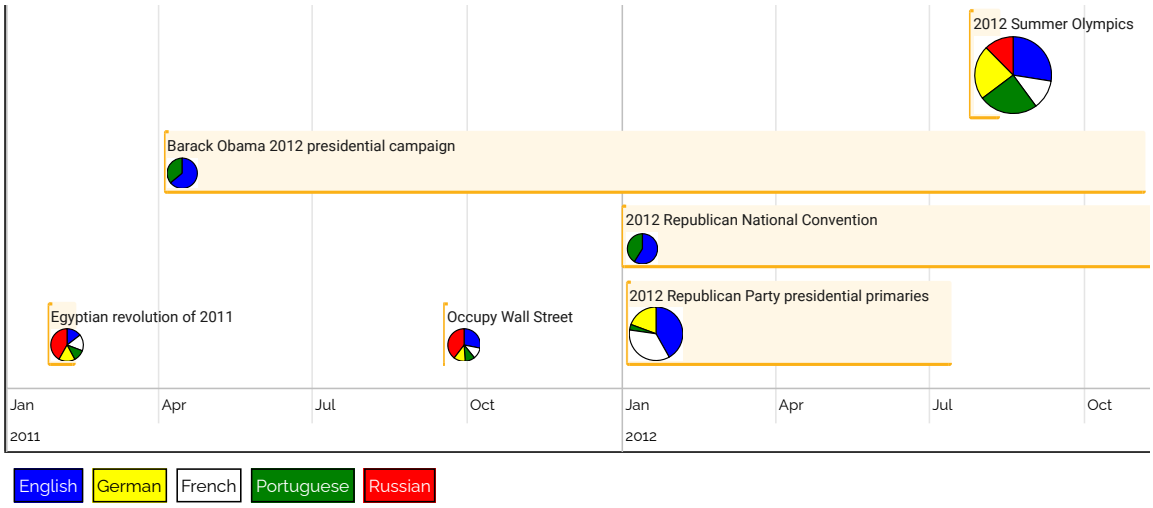


Figure 6.7. An excerpt of an *EventKG+TL* timeline representing events related to the *query entity* "2012 United States presidential election" in the time interval 01/2011-12/2013, overall including the top-10 events for five available language contexts ranked according to rc_3 – i.e., a combination of the popularity and the relation strength of the events towards the elections. Each event is represented by a labelled pie chart. The size of the pie chart corresponds to the language independent event relevance according to rc_3. The coloured slices determine the ratio of the relevance in a language context (see the legend for the colour encoding). The duration of events that lasted for more than a day is marked by a yellow interval. Upon click on a timeline entry, detailed information, including scores and link counts, is shown.

Event and Relation Retrieval

To retrieve relevant information from *EventKG*, *EventKG+TL* adopts SPARQL queries. First, *EventKG+TL* retrieves the *query entity* q, including its existence time, if available. Second, *EventKG+TL* retrieves a set of events $E' \subset E$ that are connected to q via an *EventKG* relation as the subject or the object, along with the time information associated with these events. Third, the interlinking information related to the events in E' is retrieved from *EventKG*'s link relations and their `eventKG-s:links` and `eventKG-s:mentions` property values.

Event Ranking and Timeline Creation

The top-k events related to q are selected according to the ranking criterion. For each event $e \in E'$ and language $l \in L'$, the language-specific relevance score is computed using the interlinking information provided by *EventKG*. The following link counts are used:

- $count_{links}(e, l)$: Event link count, i.e., the number of links pointing to the event

e in a language context l (via `eventKG-s:links`).

- $count_{pair}(q, e, l)$: Pair count, i.e., the number of links from q to e plus the number of links from e to q in l, denoted by `eventKG-s:links` values.
- $count_{mentions}(q, e, l)$: Mention count, i.e., the number of sentences in a language context l that jointly link to q and e, denoted by `eventKG-s:mentions`.

Each count is normalised to $[0, 1]$ by dividing its value by the highest value of this count related to the events in E' in the respective language. That way, the bias resulting from the differences in the language-specific coverage is reduced. To avoid the domination of the disproportionately often linked events (e.g., the World War II), a smoothing parameter α, experimentally set to 0.25, is adopted. The scores are computed as follows:

$$\text{popularity}(e, l) = \left(\frac{count_{links}(e, l)}{max\{count_{links}(e', l) | e' \in E'\}} \right)^{\alpha} \tag{6.1}$$

$$\begin{aligned} \text{relation strength}(q, e, l) = \frac{1}{2} \cdot \left(\frac{count_{pair}(q, e, l)}{max\{count_{pair}(q, e', l) | e' \in E'\}} \right)^{\alpha} \\ + \frac{1}{2} \cdot \left(\frac{count_{mentions}(q, e, l)}{max\{count_{mentions}(q, e', l) | e' \in E'\}} \right)^{\alpha} \end{aligned} \tag{6.2}$$

The *combined* score (rc_3) is computed as a linear combination of the two ranking criteria. We experimentally set its weight to $w = 1/3$.

$$\begin{aligned} \text{combined}(q, e, l) = w \cdot \text{popularity}(e, l) \\ + (1 - w) \cdot \text{relation strength}(q, e, l) \end{aligned} \tag{6.3}$$

The resulting timeline consists of a chronologically ordered list of the top-k highest ranked events per language with respect to the ranking criterion.

System Implementation

The *EventKG+TL* system is accessible as an HTML5 website. It is implemented using the Java Spark web framework[15]. The timeline is visualized through the browser-based Javascript library vis.js[16], the pie charts are created using the Google Charts Javascript library[17], and pop-ups showing detailed event information are based on Twitter Bootstrap[18].

[15] http://sparkjava.com/
[16] http://visjs.org/timeline_examples.html
[17] https://developers.google.com/chart/interactive/docs/gallery/piechart
[18] https://getbootstrap.com/

6.9 Discussion

In this chapter, we introduced two demonstrators which allow the interaction with *EventKG* without expertise of *EventKG* or knowledge graphs in general. A special focus was on the problem of biography timeline generation. In order to generate biography timelines from a large-scale temporal knowledge graph, we proposed a method based on distant supervision. This method uses features extracted from the temporal knowledge graph as well as a benchmark extracted from external biographical sources to train an effective relevance model. Our results of a user study and an automatic evaluation demonstrate the effectiveness of the proposed method. Our method significantly outperforms the baseline in the biography generation. According to the rater preference score, our method achieves 68% on average, in contrast to the baseline that achieves only 14%.

We presented *EventKG+BT* that generates a concise overview of a person's biography on an interactive timeline from *EventKG*. Thus, *EventKG+BT* demonstrates how knowledge graphs can facilitate research on notable accomplishments and essential events in the life of people of public interest.

We also presented *EventKG+TL* that generates event timelines, with a specific focus on the language-specific relevance of events in the surroundings of the event of user interest.

Knowledge graph applications for non-expert end-users need to take the step from using SPARQL queries for accessing the knowledge graph towards interfaces which add a layer of abstraction to the knowledge graph in a way that a user does not even need to know about the existence of a knowledge graph below the surface. *EventKG+BT* and *EventKG+TL* are two examples of such interactive interfaces, and while they provide visualisations of specific aspects in an event knowledge graph, there are still many options for extension or the creation of new interfaces. One may think that an ideal system based on a knowledge graph allows access to the whole complexity of a knowledge graph plus machine learning models applied to it. However, our systems illustrate that the selection of particular sub-tasks already serves as a great entry point towards exploring event knowledge.

7 Conclusions and Future Work

World knowledge that covers people, places, their histories and cultures, and much more keeps on growing: Every day, new events are happening which may impact the world as a whole or large communities across the globe. To grasp this knowledge, to make it accessible and understandable, there is a need to accumulate such knowledge in a way that is both machine-readable but still retains the semantics behind all involved concepts. One solution towards this goal is the creation and use of knowledge graphs, where all the concepts are nodes in a graph, connected if they are related to each other. Current knowledge graphs only insufficiently model and cover events and temporal relations [FEMR15]. Therefore, I set as my goal to create a novel event knowledge graph, to enrich it and to make it accessible to any user via interactive applications.

7.1 Summary of Contributions

In this thesis, I have dealt with knowledge graphs, with a special focus on events. Lead by four research questions introduced in Chapter 1, I followed a pipeline consisting of three steps: (i) knowledge graph creation, (ii) knowledge graph enrichment and (iii) the application of knowledge graphs.

7.1.1 Knowledge Graph Creation

I have presented two approaches for representing knowledge as a graph: (i) event knowledge graph creation from (semi-)structured sources and (ii) knowledge graph creation from tabular data using background knowledge.

Event Knowledge Graph Creation

In Chapter 3, I presented *EventKG*, my temporal and event-centric knowledge graph. I have defined a schema and an extraction pipeline, which used data from several sources: Wikidata, YAGO and several DBpedia language editions, as well as multilingual text data proceeded from Wikipedia and Wikipedia's Current Events Portal. The manual evaluation showed that the fusion of event times and event locations from different sources through rules and majority voting was successful in 75% and 94% of the cases with conflicting information from the sources (compared to 54% and 96% in the case of Wikidata, which has fewer event locations, though).

In its current version, *EventKG* V3.0 provides information for over 1.3 million events and over 4.5 million temporal relations, far more than any of its sources. *EventKG* is an extensible event-centric resource modelled in RDF that relies on Open Data and best practices to make event data spread across different sources available through a common representation. *EventKG* is reusable for a variety of novel algorithms and real-world applications, including Question Answering [CGD20], image classification [MBSH+21], recommendation [AGD20] and news analytics [MBTD+21]. *EventKG* is at the core of the Open Event Knowledge Graph [GKA+21] and has been cited more than 75 times until now [GD18a, GD19a].

Knowledge Graph Creation from Tabular Data

During the creation of *EventKG*, I assumed control over the data contained in the sources. In Chapter 4, I introduced *Tab2KG* where I tackle an opposite scenario: given a data table and domain background knowledge, I transform the data into a new knowledge graph, without knowing any semantics beforehand, i.e., without any additional user input or a user-defined extraction pipeline. To this end, I have first defined semantic profiles, which reflect the background knowledge of a specific domain (e.g., weather data). Based on these profiles, I trained a Siamese neural network that rates the similarity between a semantic class contained in the domain profile and the values contained in a table column. Using a graph-based algorithm, I have identified the relations between those semantic classes. Put together, the identification of semantic classes and relations enabled us to create a knowledge graph from the data tables. *Tab2KG* outperforms an embedding baseline by 9 percentage points on five datasets.

7.1.2 Knowledge Graph Enrichment

Under the Open-World Assumption, a knowledge graph is incomplete by nature. This calls for knowledge graph enrichment methods. I have introduced a novel approach called *HapPenIng* to enrich event knowledge graphs in Chapter 5. In contrast to existing enrichment methods [Pau17], *HapPenIng* does not only add new edges to the

knowledge graph, but it does also create new nodes – without the use of any external knowledge. To do so, I rely on a specific type of nodes that lies in event knowledge graphs: event series such as the Wimbledon Championships.

For event series completion in an event knowledge graph, I proposed two steps: First, I trained machine learning models to predict missing sub-event relations. Second, I ran a graph-based algorithm for the detection of missing event series editions under the assumption of similar patterns within event series. For determining when a new node is to be added, I created a set of constraints. As a result of *HapPenIng*, new nodes are added to the knowledge graph and enriched with a label, locations and a time span. These event characteristics are inferred using a rule-based approach and a label generation algorithm based on edit distances.

For the first step of sub-event relation prediction, I trained a random forest classifier which has an accuracy of 0.98 in 10-fold cross validation. The second step of event inference was evaluated as follows: Under the most balanced constraint, I could reconstruct close to half of randomly removed event nodes, with a precision of 0.70 – outperforming an embedding-based baseline with a precision of 0.26. All together, I created a dataset of 90,000 new sub-event relations and over 5,000 events missing in Wikidata.

7.1.3 Knowledge Graph Applications

Representing and storing knowledge alone does not imply access to it and understanding of it. Therefore, I took the last crucial step of creating applications which enable non-expert users to explore my event-centric knowledge graph. In Chapter 6, I demonstrated two systems:

- In Section 6.7, I introduced *EventKG+BT*. *EventKG+BT* is a system that let's a user explore the lives of any persons that are represented in *EventKG*. Instead of making users read a whole biography text about a person of interest, they can easily interact with *EventKG+BT* and follow what was really important in that person's life. At the example of Barack Obama, I demonstrated how this system facilitates a fast overview of his life, including the most relevant events and temporal relations.

- In Section 6.8, I introduced *EventKG+TL*. *EventKG+TL* is a system that let's a user explore any event of interest that is represented in *EventKG*. I have defined several criteria for rating the relevance of an event based on its popularity and its relation strength towards other events. Based on these criteria, I have created language-specific relevance scores which are directly reflected by the interactive tool that I created. At the example of the Brexit, I showed relevant events in its surroundings and how their relevance towards the Brexit varied across languages.

Both systems take a step towards the reduction of the workload that is necessary when closely reading encyclopedic articles, which is a significant aspect in event analytics as I have found in my prior research [GBRD18].

In Chapter 6, i also describe my approach for the creation of biography timelines. These timelines are of immediate benefit to end-users and serve as a basis for *EventKG+BT*. In a first step, I have created a publicly available benchmark for training and evaluating biography timelines by mapping temporal relations found in *EventKG* to textual biographies. I trained a classifier on this benchmark and on features extracted from *EventKG*, to identify temporal relations relevant to a biography timeline. My evaluation showed that users prefer timelines created with my approach over the timelines created by the state-of-the-art Time Machine approach [ADM+15] in close to 70% of the cases. I also identified which features are particularly important for timeline creation. Finally, I have shown that *EventKG* serves as the best source for biography-relevant facts: *EventKG* contains 55% of the facts extracted from my encyclopedic benchmark, in contrast to other knowledge graphs such as YAGO and Wikidata, which only cover less than 40% of these facts.

7.2 Open Research Directions

Based on the work presented in this thesis, there is great potential for continuation.

Extensions of *EventKG*

In its current version, *EventKG* covers more than 1.3 million events in 15 languages, extracted from other knowledge graphs and event lists in Wikipedia. While *EventKG* already opens up many application scenarios, there is still potential for an extension. Examples of such extensions include but are not limited to the following four aspects:

(i) **Reduction of cultural bias:** Even though Wikipedia and the popular knowledge graphs such as Wikidata and YAGO claim to be multilingual, investigations have shown biases: research on Wikipedia has revealed a *linguistic point of view* [MS11] and a Eurocentric bias in Wikipedia [SLW+17]. Wikidata has a problem with language maldistribution [KPV+17]. Consideration of events and entities from all over the world is a requirement for an event knowledge graph not biased towards selected cultures or communities. (ii) **Live updates:** Until now, a new version of *EventKG* is built by running its extraction pipeline as a whole, on the current dumps of its sources. Therefore, *EventKG* is not suited for exploring ongoing or very recent events. Approaches of live knowledge graph updates have been studied [Pra19] and could be adopted to *EventKG*. (iii) **Extension of sources:** News is a natural source for event detection [DK92, KVW14, RvEV+16] and could be considered for adding more events to *EventKG*. (iv) **Reduction of recency bias:** With live updates, there comes the problem of recency bias: Wikipedia is skewed towards more recent events [SLW+17].

A focus on news sources alone would strengthen this issue.

As with *EventKG*, the implementation of such four aspects needs to follow the goal of representing events that are of significance. Consequently, there is a need to generate criteria for judging the significance of events before adding them to *EventKG*. Furthermore, all events in *EventKG* are represented using a common schema. Thus, *EventKG* calls for extensions, yet these extensions need to fit the requirements specific to my event-centric knowledge graph.

Validation of knowledge graphs

The creation of knowledge graphs is a potentially error-prone process. Therefore, I envision the inclusion of an additional step in my knowledge graph processing pipeline: knowledge graph validation, which is relevant both before and after enrichment of the knowledge graph. In this thesis, I have conducted a manual evaluation of specific aspects of a knowledge graph (i.e., the fusion of contrasting information about event happening times) and showed performance of my knowledge graph enrichment under very specific evaluation settings (i.e., the reconstruction of randomly removed nodes). Until now, dataset validation systems considered the distribution of single dataset attributes [SLS+18] or focused on specific types of relations, i.e., in the case of type assertions [Pau17]. I envision a system that performs knowledge graph validation based on inferred rules and constraints. For example, a knowledge graph validation system may infer a rule that the times of different event series editions do never intersect. Such a system could also infer constraints, given a sample of gold annotations: While the Second World War should always be typed as an event, Barack Obama should never. Finally, a knowledge graph validation system would be particularly interesting in my use case of knowledge graph creation from tabular data described in Chapter 4: Rules learnt from previously seen data could be applied on the unseen data and serve as a quality check.

Creation of a fully integrated application for the exploration of events

Several applications based on knowledge graphs have been developed, including those presented in Chapter 6. The goal of these applications is to make knowledge graph accessible to non-expert end-users. Typically, this means to hide the query language from the user. In return, the user needs to forgo the complexity and expressiveness of a query language. Instead, the application defines a specific usage scenario and identifies information assumed to be relevant. To tackle this trade-off, I envision a fully integrated system that gives access to all the knowledge represented in an event knowledge graph and lets a user explore any of the involved concepts: persons and places, their relations to each other, their spatio-temporal and language-specific characteristics, and more.

A

Curriculum Vitae

Studies

since 2015	**PhD Studies** Gottfried Wilhelm Leibniz Universität L3S Research Center. Hannover, Germany
2013–2015	**Master of Science Ø1.1** Gottfried Wilhelm Leibniz Universität Master thesis: *Analysing Language-Specific Differences in Multilingual Wikipedia*
2010–2013	**Bachelor of Science Ø1.4** Gottfried Wilhelm Leibniz Universität Bachelor thesis: *Erweiterung eines ähnlichkeitsbasierten Matchingverfahrens für räumliche Objekte um inkrementelles Matching*

Professional Experience

since 2020	**Project Leader** Simple-ML project (funded by the Federal Ministry of Education and Research, BMBF)
2019–2020	**Teaching Assistent** Introduction to Data Science
2019 & 2020	**Workshop Organisation** International Workshop on Dataset PROFILing and Search (PROFILES) at the ISWC
2018 & 2019	**Teaching Assistent** Advanced Methods of Information Retrieval
since 2018	**Work Package Leader** Simple-ML project
2016	**Research Visit** KEYSTONE Short Term Scientific Mission (STSM) at the University of Southampton, UK
since 2015	**PC Member and Reviewer** Member of programme committees (CIKM Resource, CIKM Applied Research, & ESWC In-use track, PROFILES) and (sub) reviewer for several journals and conferences
2015–2018	**Project Member** ALEXANDRIA project (European Research Council Nr. 339233)

Publications

Please refer to the foreword of this thesis for a full list of my publications.

Awards

- Best Paper Award for the paper "OEKG: The Open Event Knowledge Graph" at the Workshop on Cross-lingual Event-centric Open Analytics co-located with The Web Conference [GKA+21].

Bibliography

[ABBC+17] Mohamed Al-Badrashiny, Jason Bolton, Arun Tejasvi Chaganty, Kevin Clark, Craig Harman, Lifu Huang, Matthew Lamm, Jinhao Lei, Di Lu, Xiaoman Pan, et al. TinkerBell: Cross-lingual Cold-Start Knowledge Base Construction. In *Proceedings of the 2017 Text Analysis Conference*, 2017.

[ABK+07] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. DBpedia: A Nucleus for a Web of Open Data. In *The semantic web*, pages 722–735. Springer, 2007.

[ACHZ11] Keith Alexander, Richard Cyganiak, Michael Hausenblas, and Jun Zhao. Describing Linked Datasets with the VoID Vocabulary. https://www.w3.org/TR/void/, 2011.

[ADM+15] Tim Althoff, Xin Luna Dong, Kevin Murphy, Safa Alai, Van Dang, and Wei Zhang. TimeMachine: Timeline Generation for Knowledge-base Entities. In *Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 19–28. ACM, 2015.

[AGD20] Sara Abdollahi, Simon Gottschalk, and Elena Demidova. EventKG+Click: A Dataset of Language-specific Event-centric User Interaction Traces. In *Proceedings of the CLEOPATRA Workshop at the 17th Extended Semantic Web Conference*, 2020.

[AGN15] Ziawasch Abedjan, Lukasz Golab, and Felix Naumann. Profiling Relational Data: a Survey. *The VLDB Journal*, 24(4):557–581, 2015.

[AKC19] Ahmad Alobaid, Emilia Kacprzak, and Oscar Corcho. Typology-based Semantic Labeling of Numeric Tabular Data. *Semantic Web*, 2019.

[Alb18] Frederike Albrecht. Natural Hazard Events and Social Capital: the Social Impact of Natural Disasters. *Disasters*, 42(2):336–360, 2018.

[APL98] James Allan, Ron Papka, and Victor Lavrenko. On-Line New Event Detection and Tracking. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '98, pages 37–45, New York, NY, USA, 1998. ACM.

[ARP17] Albin Ahmeti, Simon Razniewski, and Axel Polleres. Assessing the Completeness of Entities in Knowledge Bases. In *Proceedings of the 14th Extended Semantic Web Conference*, pages 7–11. Springer, 2017.

[AS13] Omar Alonso and Kyle Shiells. Timelines as Summaries of Popular Scheduled Events. In *Proceedings of the 22nd International Conference on World Wide Web*, pages 1037–1044, 2013.

[BEBB+18] Mohamed Ben Ellefi, Zohra Bellahsene, John G Breslin, Elena Demidova, Stefan Dietze, Julian Szymański, and Konstantin Todorov. RDF Dataset Profiling–a Survey of Features, Methods, Vocabularies and Applications. *Semantic Web Journal*, pages 1–29, 2018.

[BEP+08] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a Collaboratively Created Graph Database for Structuring Human Knowledge. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 1247–1250, 2008.

[BF00] Tim Berners-Lee and Mark Fischetti. *Weaving the Web - the Original Design and Ultimate Destiny of the World Wide Web by its Inventor*. HarperBusiness, 2000.

[BGM14] Dan Brickley, Ramanathan V Guha, and Brian McBride. RDF Schema 1.1. *W3C recommendation*, 25:2004–2014, 2014.

[BL06] Tim Berners-Lee. Linked Data. W3C Design Issues. https://www.w3.org/DesignIssues/LinkedData.html, 2006.

[BLHL01] Tim Berners-Lee, James Hendler, and Ora Lassila. The Semantic Web. *Scientific american*, 284(5):34–43, 2001.

[BLO+15] Elizabeth Boschee, Jennifer Lautenschlager, Sean O'Brien, Steve Shellman, James Starz, and Michael Ward. ICEWS Coded Event Data. *Harvard Dataverse*, 12, 2015.

[BRN18] Vevake Balaraman, Simon Razniewski, and Werner Nutt. Recoin: Relative Completeness in Wikidata. In *Companion Proceedings of the 2018 World Wide Web Conference*, pages 1787–1792, 2018.

[BUGD+13] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating Embeddings for Modeling Multi-relational Data. In *Proceedings of the 26th Conference on Advances in Neural Information Processing Systems*, pages 2787–2795, 2013.

[C+14] World Wide Web Consortium et al. Data catalog vocabulary (dcat), 2014.

[CCDPP19] Vincenzo Cutrona, Michele Ciavotta, Flavio De Paoli, and Matteo Palmonari. ASIA: a Tool for Assisted Semantic Interpretation and Annotation of Tabular Data. In *Proceedings of the 16th International Semantic Web Conference (Posters & Demos)*, 2019.

[CDPRS20] Marco Cremaschi, Flavio De Paoli, Anisa Rula, and Blerina Spahiu. A Fully Automated Approach to a Complete Semantic Table Interpretation. *Future Generation Computer Systems*, 2020.

[CGD20] Tarcísio Souza Costa, Simon Gottschalk, and Elena Demidova. Event-QA: A Dataset for Event-Centric Question Answering over Knowledge Graphs. In *Proceedings of the 29th Conference on Information and Knowledge Management*, 2020.

[Che17] Melisachew Wudage Chekol. Scaling Probabilistic Temporal Query Evaluation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 697–706. ACM, 2017.

[CHL+18] Michael Cafarella, Alon Halevy, Hongrae Lee, Jayant Madhavan, Cong Yu, Daisy Zhe Wang, and Eugene Wu. Ten Years of Webtables. *Proceedings of the VLDB Endowment International Conference on Very Large Data Bases*, 11(12):2140–2149, 2018.

[CJHD18] Zhiyu Chen, Haiyan Jia, Jeff Heflin, and Brian D Davison. Generating Schema Labels through Dataset Content Analysis. In *Companion Proceedings of the The Web Conference 2018*, pages 1515–1522, 2018.

[CJRHS19] Jiaoyan Chen, Ernesto Jiménez-Ruiz, Ian Horrocks, and Charles Sutton. ColNet: Embedding the Semantics of Web Tables for Column Type Prediction. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, pages 29–36, 2019.

[CKN+20] Shuang Chen, Alperen Karaoglu, Carina Negreanu, Tingting Ma, Jin-Ge Yao, Jack Williams, Andy Gordon, and Chin-Yew Lin. Linkingpark: An integrated approach for semantic table interpretation. In Ernesto Jiménez-Ruiz, Oktie Hassanzadeh, Vasilis Efthymiou, Jiaoyan Chen, Kavitha Srinivas, and Vincenzo Cutrona, editors, *Proceedings of the Semantic Web Challenge on Tabular Data to Knowledge Graph Matching (SemTab 2020) co-located with the 19th International Semantic Web Conference (ISWC 2020), Virtual conference (originally planned to be in Athens, Greece), November 5, 2020*, volume 2775 of *CEUR Workshop Proceedings*, pages 65–74. CEUR-WS.org, 2020.

[Con08] World Wide Web Consortium. SPARQL Query Language for RDF. https://www.w3.org/TR/rdf-sparql-query/, 2008.

[CPCF20] Oscar Corcho, Freddy Priyatna, and David Chaves-Fraga. Towards a New Generation of Ontology Based Data Access. *Semantic Web*, pages 1–8, 2020.

[CPSS17] Melisachew Wudage Chekol, Giuseppe Pirrò, Joerg Schoenfisch, and Heiner Stuckenschmidt. Marrying Uncertainty and Time in Knowledge Graphs. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, pages 88–94, 2017.

[CRH17] Andrew Chisholm, Will Radford, and Ben Hachey. Learning to Generate One-sentence Biographies from Wikidata. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, 2017.

[CRSDP19] Marco Cremaschi, Anisa Rula, Alessandra Siano, and Flavio De Paoli. MantisTable: A Tool for Creating Semantic Annotations on Tabular Data. In *Proceedings of the 16th Extended Semantic Web Conference*, pages 18–23. Springer, 2019.

[CZC18] Hongyun Cai, Vincent W Zheng, and Kevin Chen-Chuan Chang. A Comprehensive Survey of Graph Embedding: Problems, Techniques, and Applications. *IEEE Transactions on Knowledge and Data Engineering*, 30(9):1616–1637, 2018.

[DGH+14] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. Knowledge Vault: A Web-Scale Approach to Probabilistic Knowledge Fusion. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 601–610, 2014.

[DGH+15] Xin Luna Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Kevin Murphy, Shaohua Sun, and Wei Zhang. From Data Fusion to Knowledge Fusion. *Proceedings of the VLDB Endowment International Conference on Very Large Data Bases*, 2015.

[Dic89] Oxford English Dictionary. Oxford English Dictionary. *Simpson, JA & Weiner, ESC*, 1989.

[DJ10] Margaret Deery and Leo Jago. Social Impacts of Events and the Role of Anti-social Behaviour. *International Journal of Event and Festival Management*, 2010.

[DK92] Daniel Dayan and Elihu Katz. *Media Events*. Harvard University Press, 1992.

[Doe03] Martin Doerr. The CIDOC Conceptual Reference Module: an Ontological Approach to Semantic Interoperability of Metadata. *AI magazine*, 24(3):75–75, 2003.

[DON13] Elena Demidova, Iryna Oelze, and Wolfgang Nejdl. Aligning Freebase with the YAGO Ontology. In *Proceedings of the 22nd ACM International Conference on Information and Knowledge Management*, pages 579–588. ACM, 2013.

[DPRN17] Fariz Darari, Eko Prasojo, Simon Razniewski, and Werner Nutt. COOL-WD: A Completeness Tool for Wikidata. In *Proceedings of the 16th International Semantic Web Conference*, 2017.

[DVSC+14] Anastasia Dimou, Miel Vander Sande, Pieter Colpaert, Ruben Verborgh, Erik Mannens, and Rik Van de Walle. RML: A Generic Language for Integrated RDF Mappings of Heterogeneous Data. In *Proceedings of the 7th Workshop on Linked Data on the Web*, 2014.

[DZN12] Elena Demidova, Xuan Zhou, and Wolfgang Nejdl. FreeQ: An Interactive Query Interface for Freebase. In *Proceedings of the 21st World Wide Web Conference*, pages 325–328, 2012.

[DZN13] Elena Demidova, Xuan Zhou, and Wolfgang Nejdl. Efficient Query Construction for Large Scale Data. In *Proceedings of the 36th International ACM Conference on Research and Development in Information Retrieval*, pages 573–582, 2013.

[EDG+17] Hady Elsahar, Elena Demidova, Simon Gottschalk, Christophe Gravier, and Frederique Laforest. Unsupervised Open Relation Extraction. In *Proceedings of the 14th Extended Semantic Web Conference*, pages 12–16. Springer, 2017.

[EDT] Extended Date/Time Format (EDTF) Specification. The Library of Congress.

[EGK+14] Fredo Erxleben, Michael Günther, Markus Krötzsch, Julian Mendez, and Denny Vrandečić. Introducing Wikidata to the Linked Data Web. In *Proceedings of the 13th International Semantic Web Conference.*, pages 50–65, 2014.

[EHRMC17] Vasilis Efthymiou, Oktie Hassanzadeh, Mariano Rodriguez-Muro, and Vassilis Christophides. Matching Web Tables with Knowledge Base Entities: From Entity Lookups to Entity Embeddings. In *Proceedings of the 16th International Semantic Web Conference*, pages 260–277. Springer, 2017.

[Els16] Samir Elsharbaty. Histropedia: The Power of Data Visualisation Combined with Free Knowledge. *Wikimedia Blog*, 30, 2016.

[EMN+15] Diego Esteves, Diego Moussallem, Ciro Baron Neto, Tommaso Soru, Ricardo Usbeck, Markus Ackermann, and Jens Lehmann. MEX Vocabulary: a Lightweight Interchange Format for Machine Learning Experiments. In *Proceedings of the 11th International Conference on Semantic Systems*, pages 169–176, 2015.

[FBCC+10] David Ferrucci, Eric Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya A Kalyanpur, Adam Lally, J William Murdock, Eric Nyberg, John Prager, et al. Building Watson: An Overview of the DeepQA Project. *AI magazine*, 31(3):59–79, 2010.

[FEMR15] Michael Färber, Basil Ell, Carsten Menne, and Achim Rettinger. A Comparative Survey of DBpedia, Freebase, OpenCyc, Wikidata, and YAGO. *Semantic Web Journal*, 2015.

[FIND18] Pavlos Fafalios, Vasileios Iosifidis, Eirini Ntoutsi, and Stefan Dietze. TweetsKB: A Public and Large-scale RDF Corpus of Annotated Tweets. In *Proceedings of the 15th Extended Semantic Web Conference*, pages 177–190. Springer, 2018.

[Gar19] Massimiliano Garda. A Semantics-Enabled Approach for Data Lake Exploration Services. In *Proceedings of the IEEE World Congress on Services*, volume 2642, pages 327–330. IEEE, 2019.

[GBRD18] Simon Gottschalk, Viola Bernacchi, Richard Rogers, and Elena Demidova. Towards better Understanding Researcher Strategies in Cross-lingual Event Analytics. In *Proceedings of the 22nd International Conference on Theory and Practice of Digital Libraries*, 2018.

[GD16] Simon Gottschalk and Elena Demidova. Analysing Temporal Evolution of Interlingual Wikipedia Article Pairs. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1089–1092, 2016.

[GD17] Simon Gottschalk and Elena Demidova. MultiWiki: Interlingual Text Passage Alignment in Wikipedia. *TWEB*, 11(1):6:1–6:30, 2017.

[GD18a] Simon Gottschalk and Elena Demidova. EventKG: A Multilingual Event-Centric Temporal Knowledge Graph. In *Proceedings of the 15th Extended Semantic Web Conference*, pages 272–287. Springer, 2018.

[GD18b] Simon Gottschalk and Elena Demidova. EventKG+TL: Creating Cross-Lingual Timelines from an Event-Centric Knowledge Graph. In *Proceedings of the 15th Extended Semantic Web Conference*, pages 164–169, 2018.

[GD19a] Simon Gottschalk and Elena Demidova. EventKG - the Hub of Event Knowledge on the Web - and Biographical Timeline Generation. *Semantic Web*, 2019.

[GD19b] Simon Gottschalk and Elena Demidova. HapPenIng: Happen, Predict, Infer—Event Series Completion in a Knowledge Graph. In *Proceedings of the International Semantic Web Conference*, pages 200–218. Springer, 2019.

[GD20] Simon Gottschalk and Elena Demidova. EventKG+BT: Generation of Interactive Biography Timelines from a Knowledge Graph. In *Proceedings of the 17th Extended Semantic Web Conference (Satellite Events)*, 2020.

[GDBR17] Simon Gottschalk, Elena Demidova, Viola Bernacchi, and Richard Rogers. Ongoing Events in Wikipedia: A Cross-lingual Case Study. In *Proceedings of the 9th International Web Science Conference*, pages 387–388, 2017.

[GDR15] Gerhard Gossen, Elena Demidova, and Thomas Risse. iCrawl: Improving the Freshness of Web Collections by Integrating Social Web and Focused Web Crawling. In *Proceedings of the 15th ACM/IEEE-CS Joint Conference on Digital Libraries*, 2015.

[GKA+21] Simon Gottschalk, Endri Kacupaj, Sara Abdollahi, Diego Alves, Gabriel Amaral, Elisavet Koutsiana, Tin Kuculo, Daniela Major, Caio Mello, Gullal Singh Cheema, Abdul Sittar, Swati, Golsa Tahmasebzadeh, and Gaurish Thakkar. OEKG: The Open Event Knowledge Graph. In *Proceedings of the CLEOPATRA Workshop at the 30th The Web Conference*, 2021.

[GTK+19] Simon Gottschalk, Nicolas Tempelmeier, Günter Kniesel, Vasileios Iosifidis, Besnik Fetahu, and Elena Demidova. Simple-ML: Towards a Framework for Semantic Data Analytics Workflows. In *Proceedings of the 15th International Conference on Semantic Systems*, pages 359–366. Springer, 2019.

[Guh11] Ramanathan Guha. Introducing schema.org: Search Engines Come Together for a Richer Web. *Google Official Blog*, 2011.

[Har17] Olaf Hartig. Foundations of RDF* and SPARQL*: (An alternative approach to statement-level metadata in RDF). In *Proceedings of the 11th Alberto Mendelzon International Workshop on Foundations of Data Management and the Web*, volume 1912. Juan Reutter, Divesh Srivastava, 2017.

[HBC+20] Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard de Melo, Claudio Gutierrez, José Emilio Labra Gayo, Sabrina Kirrane, Sebastian Neumaier, Axel Polleres, et al. Knowledge graphs. *arXiv preprint arXiv:2003.02320*, 2020.

[Hef04] Jeff Heflin. OWL Web Ontology Language-use Cases and Requirements. *W3C Recommendation*, 10(10):1–12, 2004.

[Her16] Astrid Herbold. Happy birthday, Sorgenkind. *ZEIT ONLINE*, 2016.

[HHB+19] Madelon Hulsebos, Kevin Hu, Michiel Bakker, Emanuel Zgraggen, Arvind Satyanarayan, Tim Kraska, Çagatay Demiralp, and César Hidalgo. Sherlock: A Deep Learning Approach to Semantic Data Type Detection. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1500–1508, 2019.

[HHK+10] Andreas Harth, Katja Hose, Marcel Karnstedt, Axel Polleres, Kai-Uwe Sattler, and Jürgen Umbrich. Data Summaries for On-demand Queries over Linked Data. In Michael Rappa, Paul Jones, Juliana Freire, and Soumen Chakrabarti, editors, *Proceedings of the 19th International Conference on World Wide Web, WWW 2010, Raleigh, North Carolina, USA, April 26-30, 2010*, pages 411–420. ACM, 2010. DOI: https://doi.org/10.1145/1772690.1772733.

[HKLT19] Shou-Ching Hsiao, Da-Yu Kao, Zi-Yuan Liu, and Raylin Tso. Malware Image Classification using One-shot Learning with Siamese Networks. *Procedia Computer Science*, 159:1863–1871, 2019.

[HLC+20] Viet-Phi Huynh, Jixiong Liu, Yoan Chabot, Thomas Labbé, Pierre Monnin, and Raphaël Troncy. DAGOBAH: Enhanced Scoring Algorithms for Scalable Annotations of Tabular Data. In Ernesto Jiménez-Ruiz, Oktie Hassanzadeh, Vasilis Efthymiou, Jiaoyan Chen, Kavitha Srinivas, and Vincenzo Cutrona, editors, *Proceedings of the Semantic Web Challenge on Tabular Data to Knowledge Graph Matching (SemTab 2020) co-located with the 19th International Semantic Web Conference (ISWC 2020), Virtual conference (originally planned to be in Athens, Greece), November 5, 2020*, volume 2775 of *CEUR Workshop Proceedings*, pages 27–39. CEUR-WS.org, 2020.

[HLLS17] Carsten Hartenfels, Martin Leinberger, Ralf Lämmel, and Steffen Staab. Type-Safe Programming with OWL in Semantics4J. In *Proceedings of the 14th International Semantic Web Conference (Posters & Demos)*, 2017.

[HLT+19] Eero Hyvönen, Petri Leskinen, Minna Tamper, Heikki Rantala, Esko Ikkala, Jouni Tuominen, and Kirsi Keravuori. BiographySampo–Publishing and Enriching Biographies on the Semantic Web for Digital Humanities Research. In *Proceedings of the 16th Extended Semantic Web Conference (Satellite Events)*, pages 574–589. Springer, 2019.

[HLY19] Braden Hancock, Hongrae Lee, and Cong Yu. Generating Titles for Web Tables. In *Proceedings of The Web Conference 2019*, pages 638–647, 2019.

[HM+06] John Horne, Wolfram Manzenreiter, et al. Sports Mega-events: Social Scientific Analyses of a Global Phenomenon. *Sociological Review*, 54(Suppl. 2):1–187, 2006.

[HP06] Jerry R Hobbs and Feng Pan. Time Ontology in OWL. *W3C working draft*, 27:133, 2006.

[HQRA+13] Arvid Heise, Jorge-Arnulfo Quiané-Ruiz, Ziawasch Abedjan, Anja Jentzsch, and Felix Naumann. Scalable Discovery of Unique Column Combinations. *VLDB Endowment International Conference on Very Large Data Bases*, 7(4):301–312, 2013.

[Hum20] Bernhard Humm. Fascinating with Open Data: openArtBrowser. In *Qurator*, 2020.

[HWM+17] Konrad Höffner, Sebastian Walter, Edgard Marx, Ricardo Usbeck, Jens Lehmann, and Axel-Cyrille Ngonga Ngomo. Survey on Challenges of Question Answering in the Semantic Web. *Semantic Web*, 8(6):895–920, 2017.

[HWP12] Daniel Hienert, Dennis Wegener, and Heiko Paulheim. Automatic Classification and Relationship Extraction for Multi-Lingual and Multi-Granular Events from Wikipedia. In *Proceedings of the Workshop on Detection, Representation, and Exploitation of Events in the Semantic Web*, 2012.

[HZLL19] Xiao Huang, Jingyuan Zhang, Dingcheng Li, and Ping Li. Knowledge Graph Embedding Based Question Answering. In *Proceedings of the 12th ACM International Conference on Web Search and Data Mining*, pages 105–113. ACM, 2019.

[ISO16] Information Technology – Database Languages – SQL Multimedia and Application Packages – Part 3: Spatial. International Organization for Standardization, 2016.

[JHE+20] Ernesto Jiménez-Ruiz, Oktie Hassanzadeh, Vasilis Efthymiou, Jiaoyan Chen, Kavitha Srinivas, and Vincenzo Cutrona. Results of semtab 2020. In Ernesto Jiménez-Ruiz, Oktie Hassanzadeh, Vasilis Efthymiou, Jiaoyan Chen, Kavitha Srinivas, and Vincenzo Cutrona, editors, *Proceedings of the Semantic Web*

Challenge on Tabular Data to Knowledge Graph Matching (SemTab 2020) co-located with the 19th International Semantic Web Conference (ISWC 2020), Virtual conference (originally planned to be in Athens, Greece), November 5, 2020, volume 2775 of *CEUR Workshop Proceedings*, pages 1–8. CEUR-WS.org, 2020.

[JRHE+20] Ernesto Jiménez-Ruiz, Oktie Hassanzadeh, Vasilis Efthymiou, Jiaoyan Chen, and Kavitha Srinivas. SemTab 2019: Resources to Benchmark Tabular Data to Knowledge Graph Matching Systems. *The Semantic Web*, 2020.

[KA13] Sven R Kunze and Sören Auer. Dataset Retrieval. In *2013 IEEE 7th International Conference on Semantic Computing*, pages 1–8. IEEE, 2013.

[KL12] Andreas Kaltenbrunner and David Laniado. There is No Deadline - Time Evolution of Wikipedia Discussions. In *Proceedings of the 8th International Symposium on Wikis and Open Collaboration*, page 6. ACM, 2012.

[Kna17] Tomás Knap. Towards Odalic, a Semantic Table Interpretation Tool in the ADEQUATe Project. In *LD4IE@ ISWC*, pages 26–37, 2017.

[KPV+17] Lucie-Aimée Kaffee, Alessandro Piscopo, Pavlos Vougiouklis, Elena Simperl, Leslie Carr, and Lydia Pintscher. A Glimpse into Babel: an Analysis of Multilinguality in Wikidata. In *Proceedings of the 13th International Symposium on Open Collaboration*, pages 1–5, 2017.

[KSA+12] Craig A Knoblock, Pedro Szekely, José Luis Ambite, Aman Goel, Shubham Gupta, Kristina Lerman, Maria Muslea, Mohsen Taheriyan, and Parag Mallick. Semi-automatically Mapping Structured Sources into the Semantic Web. In *Proceedings of the 9th Extended Semantic Web Conference*, pages 375–390. Springer, 2012.

[KSSW16] Erdal Kuzey, Vinay Setty, Jannik Strötgen, and Gerhard Weikum. As Time Goes By: Comprehensive Tagging of Textual Phrases with Temporal Scopes. In *Proceedings of the 25th International conference on World Wide Web*, pages 915–925. International World Wide Web Conferences Steering Committee, 2016.

[KVW14] Erdal Kuzey, Jilles Vreeken, and Gerhard Weikum. A Fresh Look on Knowledge Bases: Distilling Named Events from News. In *Proceedings of the 23rd International Conference on Conference on Information and Knowledge Management*, pages 1689–1698, 2014.

[KZS15] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese Neural Networks for One-shot Image Recognition. In *ICML Deep Learning Workshop*, volume 2. Lille, 2015.

[Lec19] Freddy Lecue. On the Role of Knowledge Graphs in Explainable AI. *Semantic Web*, pages 1–11, 2019.

[Len02] Maurizio Lenzerini. Data Integration: A Theoretical Perspective. In *Proceedings of the 21st ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pages 233–246, 2002.

[LFBG14] Gregor Leban, Blaz Fortuna, Janez Brank, and Marko Grobelnik. Event Registry: Learning about World Events from News. In *Proceedings of the 23rd International Conference on World Wide Web*, pages 107–110, 2014.

[LGA16] Rémi Lebret, David Grangier, and Michael Auli. Neural Text Generation from Structured Data with Application to the Biography Domain. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2016.

[LIJ+15] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. DBpedia - A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia. *Semantic Web Journal*, 6(2):167–195, 2015.

[LLCZ18] Xusheng Luo, Kangqi Luo, Xianyang Chen, and Kenny Q. Zhu. Cross-Lingual Entity Linking for Web Tables. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, pages 362–369. AAAI Press, 2018.

[LS13] Kalev Leetaru and Philip A Schrodt. GDELT: Global Data on Events, Location and Tone, 1979-2012. In *ISA Annual Convention*, pages 1–49. Citeseer, 2013.

[LS18] Jonathan Lajus and Fabian M Suchanek. Are All People Married?: Determining Obligatory Attributes in Knowledge Bases. In *Proceedings of the 2018 World Wide Web Conference*, pages 1115–1124, 2018.

[LSB+17] Jens Lehmann, Gezim Sejdiu, Lorenz Bühmann, Patrick Westphal, Claus Stadler, Ivan Ermilov, Simon Bin, et al. Distributed Semantic Analytics using the SANSA Stack. In *Proceedings of the 14th International Semantic Web Conference*, pages 147–155. Springer, 2017.

[LT19] Jixiong Liu and Raphaël Troncy. DAGOBAH: An End-to-End Context-Free Tabular Data Semantic Annotation System. In *SemTab@ISWC*, 2019.

[Mat37] Dean W.R. Matthews. What Is an Historical Event? In *Proceedings of the Aristotelian Society*, pages 207–216. JSTOR, 1937.

[MB16] Arunav Mishra and Klaus Berberich. Leveraging Semantic Annotations to Link Wikipedia and News Archives. In *Proceedings of the 38th European Conference on Information Retrieval*, pages 30–42. Springer, 2016.

[MBS14] Farzaneh Mahdisoltani, Joanna Biega, and Fabian Suchanek. YAGO3: A Knowledge Base from Multilingual Wikipedias. In *Proceedings of the 8th Conference on Innovative Data Systems Research*, 2014.

[MBSH+21] Eric Müller-Budack, Matthias Springstein, Sherzod Hakimov, Kevin Mrutzek, and Ralph Ewerth. Ontology-driven Event Type Classification in Images.

In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*, 2021.

[MBTD+21] Eric Müller-Budack, Jonas Theiner, Sebastian Diering, Maximilian Idahl, Sherzod Hakimov, and Ralph Ewerth. Multimodal news analytics using measures of cross-modal entity and context consistency. *International Journal of Multimedia Information Retrieval*, pages 1–15, 2021.

[McN47] Quinn McNemar. Note on the Sampling Error of the Difference between Correlated Proportions or Percentages. *Psychometrika*, 12(2):153–157, 1947.

[MEC+18] James P McCusker, J Erickson, Katherine Chastain, Sabbir Rashid, Rukmal Weerawarana, and D McGuinness. What is a Knowledge Graph? *Semantic Web Journal*, 2018.

[MGS+19] Mohamed Nadjib Mami, Damien Graux, Simon Scerri, Hajira Jabeen, Sören Auer, and Jens Lehmann. Squerall: Virtual Ontology-Based Access to Heterogeneous and Large Data Sources. In *Proceedings of the 18th International Semantic Web Conference*, pages 229–245. Springer, 2019.

[Mil98] George A Miller. *WordNet: An Electronic Lexical Database*. MIT press, 1998.

[MJGSB11] Pablo N Mendes, Max Jakob, Andrés García-Silva, and Christian Bizer. DBpedia Spotlight: Shedding Light on the Web of Documents. In *Proceedings of the 7th International Conference on Semantic Systems*, pages 1–8. ACM, 2011.

[MKG+18] Stanislav Malyshev, Markus Krötzsch, Larry González, Julius Gonsior, and Adrian Bielefeldt. Getting the Most out of Wikidata: Semantic TechnologyUsage in Wikipedia's Knowledge Graph. In *Proceedings of the 17th International Semantic Web Conference*, pages 376–394. Springer, 2018.

[MNUP16] Johann Mitlöhner, Sebastian Neumaier, Jürgen Umbrich, and Axel Polleres. Characteristics of Open Data CSV Files. In *Proceedings of the 2nd International Conference on Open and Big Data*, pages 72–79. IEEE, 2016.

[Moo18] Paul Mooney. Kaggle Machine Learning & Data Science Survey. *kaggle*, 2018.

[MS11] Paolo Massa and Federico Scrinzi. Exploring Linguistic Points of View of Wikipedia. In *Proceedings of the 7th International Symposium on Wikis and Open Collaboration*, pages 213–214, 2011.

[MSB+14] Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. The Stanford CoreNLP Natural Language Processing Toolkit. In *Proceedings of the 52nd Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, 2014.

[MVH04] Deborah L. McGuinness and Frank Van Harmelen. OWL Web Ontology Language Overview. *W3C recommendation*, 10(10):2004, 2004.

[Mye86] Eugene W Myers. An O(ND) Difference Algorithm and its Variations. *Algorithmica*, 1(1-4):251–266, 1986.

[NKIT19] Phuc Nguyen, Natthawut Kertkeidkachorn, Ryutaro Ichise, and Hideaki Takeda. MTab: Matching Tabular Data to Knowledge Graph using Probability Models. In *Proceedings of the Semantic Web Challenge on Tabular Data to Knowledge Graph Matching co-located with the 18th International Semantic Web Conference*, 2019.

[NMW17] Finn Årup Nielsen, Daniel Mietchen, and Egon Willighagen. Scholia, Scientometrics and Wikidata. In *Proceedings of the 14th Extended Semantic Web Conference*, pages 237–259. Springer, 2017.

[NNIT19] Phuc Nguyen, Khai Nguyen, Ryutaro Ichise, and Hideaki Takeda. EmbNum+: Effective, Efficient, and Robust Semantic Labeling for Numerical Values. *New Generation Computing*, 37(4):393–427, 2019.

[NP19] Sebastian Neumaier and Axel Polleres. Enabling Spatio-Temporal Search in Open Data. *Journal of Web Semantics*, 55:21–36, 2019.

[NSQJ16] Dat Quoc Nguyen, Kairit Sirts, Lizhen Qu, and Mark Johnson. STransE: a Novel Embedding Model of Entities and Relationships in Knowledge Bases. In *Proceedings of the 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 2016.

[NUP16] Sebastian Neumaier, Jürgen Umbrich, and Axel Polleres. Automated Quality Assessment of Metadata across Open Data Portals. *Journal of Data and Information Quality*, 8(1):1–29, 2016.

[PAKS16] Minh Pham, Suresh Alse, Craig A. Knoblock, and Pedro A. Szekely. Semantic Labeling: A Domain-Independent Approach. In *Proceedings of the 13th International Semantic Web Conference (Posters & Demos)*, volume 9981, pages 446–462, 2016.

[Pau17] Heiko Paulheim. Knowledge Graph Refinement: A Survey of Approaches and Evaluation Methods. *Semantic Web*, 8(3):489–508, 2017.

[PB13] Heiko Paulheim and Christian Bizer. Type Inference on Noisy RDF Data. In *Proceedings of the 12th International Semantic Web Conference*, pages 510–525. Springer, 2013.

[Pei09] Charles Sanders Peirce. Existential Graphs, Manuscript 514, 1909.

[PKN18] Radityo Eko Prasojo, Mouna Kacimi, and Werner Nutt. StuffIE: Semantic Tagging of Unlabeled Facets Using Fine-Grained Information Extraction. In *Proceedings of the 27th International Conference on Information and Knowledge Management*, pages 467–476. ACM, 2018.

[PLC+08] Antonella Poggi, Domenico Lembo, Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Riccardo Rosati. Linking Data to Ontologies. In *Journal on Data Semantics X*, pages 133–173. Springer, 2008.

[PPK+18] André Pomp, Alexander Paulus, Andreas Kirmse, Vadim Kraus, and Tobias Meisen. Applying Semantics to Reduce the Time to Analytics within Complex Heterogeneous Infrastructures. *Technologies*, 6(3):86, 2018.

[Pra19] Sandra Praetor. DBpedia Live Restart – Getting Things Done. *DBpedia Blog*, 2019.

[Pre16] Gil Press. Cleaning Big Data: Most Time-Consuming, Least Enjoyable Data Science Task, Survey Says. *Forbes*, 2016.

[PTVS+16] Thomas Pellissier Tanon, Denny Vrandečić, Sebastian Schaffert, Thomas Steiner, and Lydia Pintscher. From Freebase to Wikidata: The Great Migration. In *Proceedings of the 25th International Conference on World Wide Web*, pages 1419–1428, 2016.

[RB17] Dominique Ritze and Christian Bizer. Matching Web Tables to DBpedia - a Feature Utility Study. *context*, 42(41):19–31, 2017.

[Rei81] Raymond Reiter. On Closed World Data Bases. In *Readings in Artificial Intelligence*, pages 119–140. Elsevier, 1981.

[RGP17] Michael Ringgaard, Rahul Gupta, and Fernando CN Pereira. SLING: A Framework for Frame Semantic Parsing. *arXiv preprint arXiv:1710.07032*, 2017.

[RLB15] Dominique Ritze, Oliver Lehmberg, and Christian Bizer. Matching HTML Tables to DBpedia. In *Proceedings of the 5th International Conference on Web Intelligence, Mining and Semantics*, pages 1–6, 2015.

[RLOB16] Dominique Ritze, Oliver Lehmberg, Yaser Oulabi, and Christian Bizer. Profiling the Potential of Web Tables for Augmenting Cross-domain Knowledge Bases. In *Proceedings of the 25th International Conference on World Wide Web*, pages 251–261, 2016.

[RMKS15] S Krishnamurthy Ramnandan, Amol Mittal, Craig A Knoblock, and Pedro Szekely. Assigning Semantic Labels to Data Sources. In *Proceedings of the 12th Extended Semantic Web Conference*, pages 403–417. Springer, 2015.

[Rog13] Richard Rogers. *Digital Methods*. MIT Press, 2013.

[RPN+14] Anisa Rula, Matteo Palmonari, Axel-Cyrille Ngonga Ngomo, Daniel Gerber, Jens Lehmann, and Lorenz Bühmann. Hybrid Acquisition of Temporal Scopes for RDF Data. In *Proceedings of the 11th European Semantic Web Conference*, pages 488–503. Springer, 2014.

[RS04] Laura Elena Raileanu and Kilian Stoffel. Theoretical Comparison between the Gini Index and Information Gain Criteria. *Annals of Mathematics and Artificial Intelligence*, 41(1):77–93, 2004.

[RSN16] Simon Razniewski, Fabian Suchanek, and Werner Nutt. But What Do We Actually Know? In *Proceedings of the 5th Workshop on Automated Knowledge Base Construction*, pages 40–44, 2016.

[RvEV+16] Marco Rospocher, Marieke van Erp, Piek Vossen, Antske Fokkens, Itziar Aldabe, German Rigau, Aitor Soroa, Thomas Ploeger, and Tessel Bogaard. Building Event-centric Knowledge Graphs from News. *Web Semantics*, 37:132–151, 2016.

[SA00] Russell Swan and James Allan. Automatic Generation of Overview Timelines. In *Proceedings of the 23rd International ACM Conference on Research and Development in Information Retrieval*, pages 49–56. ACM, 2000.

[SAMA17] Vinay Setty, Abhijit Anand, Arunav Mishra, and Avishek Anand. Modeling Event Importance for Ranking Daily News Events. In *Proceedings of the 10th International Conference on Web Search and Data Mining*, pages 231–240. ACM, 2017.

[Sch12] Philip A Schrodt. CAMEO: Conflict and Mediation Event Observations Event and Actor Codebook, 2012.

[SG10] Jannik Strötgen and Michael Gertz. HeidelTime: High Quality Rule-based Extraction and Normalization of Temporal Expressions. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 321–324, 2010.

[Sin12] Amit Singhal. Introducing the Knowledge Graph: Things, not Strings. *Official Google Blog*, 5, 2012.

[SKW07] Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. YAGO: A Core of Semantic Knowledge. In *Proceedings of the 16th International Conference on World Wide Web*, pages 697–706, 2007.

[SLS+18] Sebastian Schelter, Dustin Lange, Philipp Schmidt, Meltem Celikel, Felix Biessmann, and Andreas Grafberger. Automating Large-scale Data Quality Verification. *Proceedings of the VLDB Endowment International Conference on Very Large Data Bases*, 11(12):1781–1794, 2018.

[SLW+17] Anna Samoilenko, Florian Lemmerich, Katrin Weller, Maria Zens, and Markus Strohmaier. Analysing Timelines of National Histories across Wikipedia Editions: A Comparative Computational Approach. In *Eleventh International AAAI Conference on Web and Social Media*, 2017.

[SNA16] Jaspreet Singh, Wolfgang Nejdl, and Avishek Anand. History by Diversity: Helping Historians Search News Archives. In *Proceedings of the 1st Conference on Human Information Interaction and Retrieval*, pages 183–192. ACM, 2016.

[Sow76] John F Sowa. Conceptual Graphs for a Data Base Interface. *IBM Journal of Research and Development*, 20(4):336–357, 1976.

[Sow12] John F Sowa. Semantic Networks. *Encyclopedia of Cognitive Science*, 2012.

[SRLJ19] Gezim Sejdiu, Anisa Rula, Jens Lehmann, and Hajira Jabeen. A Scalable Framework for Quality Assessment of RDF Datasets. In *Proceedings of the 16th International Semantic Web Conference*, pages 261–276. Springer International Publishing, 2019.

[STH09] R. Shaw, R. Troncy, and L. Hardman. LODE: Linking Open Descriptions of Events. In *Proceedings of the 4th Asian Semantic Web Conference*, pages 153–167, 2009.

[Sur13] Mihai Surdeanu. Overview of the TAC2013 Knowledge Base Population Evaluation: English Slot Filling and Temporal Slot Filling. In *Proceedings of the 2013 Text Analysis Conference*, 2013.

[SVDTO19] Bram Steenwinckel, Gilles Vandewiele, Filip De Turck, and Femke Ongenae. CSV2KG: Transforming Tabular Data into Semantic Knowledge. In *Proceedings of the Semantic Web Challenge on Tabular Data to Knowledge Graph Matching co-located with the 18th International Semantic Web Conference*, 2019.

[SW17] Baoxu Shi and Tim Weninger. ProjE: Embedding Projection for Knowledge Graph Completion. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, pages 1236–1242, 2017.

[SWW+15] Jaeho Shin, Sen Wu, Feiran Wang, Christopher De Sa, Ce Zhang, and Christopher Ré. Incremental Knowledge Base Construction using DeepDive. *Proceedings of the VLDB Endowment International Conference on Very Large Data Bases*, 8(11):1310, 2015.

[TA14] Giang Binh Tran and Mohammad Alrifai. Indexing and Analyzing Wikipedia's Current Events Portal, the Daily News Summaries by the Crowd. In *Proceedings of the 23rd International Conference on World Wide Web*, pages 511–516. ACM, 2014.

[TAH15] Giang Tran, Mohammad Alrifai, and Eelco Herder. Timeline Summarization from Relevant Headlines. In *Proceedings of the 37th European Conference on Information Retrieval*, pages 245–256. Springer, 2015.

[TEPW11] Tran Anh Tuan, Shady Elbassuoni, Nicoleta Preda, and Gerhard Weikum. CATE: Context-aware Timeline for Entity Illustration. In *Proceedings of the 20th International Conference Companion on World Wide Web*, pages 269–272. ACM, 2011.

[TKSA16a] Mohsen Taheriyan, Craig A Knoblock, Pedro Szekely, and José Luis Ambite. Learning the Semantics of Structured Data Sources. *Journal of Web Semantics*, 37:152–169, 2016.

[TKSA16b] Mohsen Taheriyan, Craig A Knoblock, Pedro Szekely, and José Luis Ambite. Leveraging Linked Data to Discover Semantic Relations within Data Sources. In *Proceedings of the 15th International Semantic Web Conference*, pages 549–565. Springer, 2016.

[TLR16] Andreas Thalhammer, Nelia Lasierra, and Achim Rettinger. LinkSUM: Using Link Analysis to Summarize Entity Data. In *Proceedings of the 16th International Conference on Web Engineering*, pages 244–261. Springer, 2016.

[TPS+17] Thomas Tanon Pellissier, Daria Stepanova, et al. Completeness-aware Rule Learning from Knowledge Graphs. In *Proceedings of the 16th International Semantic Web Conference*, pages 507–525. Springer, 2017.

[TWM12] Partha Pratim Talukdar, Derry Wijaya, and Tom Mitchell. Coupled Temporal Scoping of Relational Facts. In *Proceedings of the 6th ACM International Conference on Web Search and Data Mining*, pages 73–82. ACM, 2012.

[TWS20] Thomas Pellissier Tanon, Gerhard Weikum, and Fabian Suchanek. YAGO 4: A Reason-able Knowledge Base. In *Proceedings of the 17th Extended Semantic Web Conference*, pages 583–596. Springer, 2020.

[VCK15] Piek Vossen, Tommaso Caselli, and Yiota Kontzopoulou. Storylines for Structuring Massive Streams of News. In *Proceedings of the First Workshop on Computing News Storylines*, pages 40–49, 2015.

[VHMS+11] Willem Robert Van Hage, Véronique Malaisé, Roxane Segers, Laura Hollink, and Guus Schreiber. Design and Use of the Simple Event Model (SEM). *Web Semantics*, 9(2):128–136, 2011.

[VK14] Denny Vrandečić and Markus Krötzsch. Wikidata: A Free Collaborative Knowledgebase. *Communications of the ACM*, 57(10):78–85, 2014.

[VKV+06] Max Völkel, Markus Krötzsch, Denny Vrandecic, Heiko Haller, and Rudi Studer. Semantic Wikipedia. In *Proceedings of the 15th International Conference on World Wide Web*, pages 585–594, 2006.

[VMdR17] Nikos Voskarides, Edgar Meij, and Maarten de Rijke. Generating Descriptions of Entity Relationships. In *Proceedings of the 39th European Conference on Information Retrieval*, pages 317–330. Springer, 2017.

[Vra20] Denny Vrandečić. Architecture for a Multilingual Wikipedia. *arXiv preprint arXiv:2004.04733*, 2020.

[W3C06] W3C. Resource Description Framework (RDF). https://www.w3.org/RDF/, 2006.

[WDA+16] Mark D Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, et al. The FAIR Guiding Principles for Scientific Data Management and Stewardship. *Scientific data*, 3(1):1–9, 2016.

[WDRS20] Gerhard Weikum, Luna Dong, Simon Razniewski, and Fabian Suchanek. Machine Knowledge: Creation and Curation of Comprehensive Knowledge Bases. *arXiv preprint arXiv:2009.11564*, 2020.

[WFHP16] Ian H Witten, Eibe Frank, Mark A Hall, and Christopher J Pal. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 2016.

[WKGS19] Qifan Wang, Bhargav Kanagal, Vijay Garg, and D Sivakumar. Constructing a Comprehensive Events Database from the Web. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 229–238, 2019.

[WMWG17] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. Knowledge Graph Embedding: A Survey of Approaches and Applications. *IEEE TKDE*, 29(12):2724–2743, 2017.

[WR09] Ryen W White and Resa A Roth. Exploratory Search: Beyond the Query-response Paradigm. *Synthesis Lectures on Information Concepts, Retrieval, and Services*, 1(1):1–98, 2009.

[WT10] Gerhard Weikum and Martin Theobald. From Information to Knowledge: Harvesting Entities and Relationships from Web Sources. In *Proceedings of the 29th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pages 65–76, 2010.

[XCK+18] Guohui Xiao, Diego Calvanese, Roman Kontchakov, Domenico Lembo, Antonella Poggi, Riccardo Rosati, and Michael Zakharyaschev. Ontology-based Data Access: A Survey. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, 2018.

[YRH+16] Amy Zhao Yu, Shahar Ronen, Kevin Hu, Tiffany Lu, and César A Hidalgo. Pantheon 1.0, a Manually Verified Dataset of Globally Famous Biographies. *Scientific data*, 3(1):1–16, 2016.

[YRH+18] Quan Yuan, Xiang Ren, Wenqi He, Chao Zhang, Xinhe Geng, Lifu Huang, Heng Ji, Chin-Yew Lin, and Jiawei Han. Open-Schema Event Profiling for Massive News Corpora. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 587–596. ACM, 2018.

[ZCZ+17] Weiguo Zheng, Hong Cheng, Lei Zou, Jeffrey Xu Yu, and Kangfei Zhao. Natural Language Question/Answering: Let Users Talk With The Knowledge Graph. In *Proceedings of the 26th Conference on Information and Knowledge Management*, pages 217–226, 2017.

[ZDFB12] Jian Zhao, Steven M Drucker, Danyel Fisher, and Donald Brinkman. TimeSlice: Interactive Faceted Browsing of Timeline Data. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, pages 433–436. ACM, 2012.

[Zha17] Ziqi Zhang. Effective and Efficient Semantic Table Interpretation using TableMiner+. *Semantic Web*, 8(6):921–957, 2017.

[ZMBR20] Shuo Zhang, Edgar Meij, Krisztian Balog, and Ridho Reinanda. Novel Entity Discovery from Web Tables. In *Proceedings of The Web Conference 2020*, pages 1298–1308, 2020.