WIRTSCHAFTSUNIVERSITÄT WIEN
Vienna University of Economics and Business

Bachelor's Thesis

Title of Bachelor's Thesis (English)	Semi-Automatic Knowledge Graph Creation and Evaluation
Title of Bachelor's Thesis (German)	Halbautomatische Erzeugung und Evaluierung von Knowledge Graphs
Author (last name, first name):	HIESINGER Veronika
Student ID number:	12024031
Degree program:	Bachelor of Science (WU), BSc (WU)
Examiner (degree, first name, last name):	Stefani Tsaneva, Prof. Dr. Marta Sabou

I hereby declare:

1. I have written this Bachelor's thesis myself, independently and without the aid of unfair or unauthorized resources. Whenever content has been taken directly or indirectly from other sources, this has been indicated and the source referenced.
2. This Bachelor's Thesis has not been previously presented as an examination paper in this or any other form in Austria or abroad.
3. This Bachelor's Thesis is identical with the thesis assessed by the examiner.
4. (Only applicable if the thesis was written by more than one author): this Bachelor's thesis was written together with

   The individual contributions of each writer as well as the co-written passages have been indicated.

13/10/2024
Date

Signature

WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

Bachelor Thesis

Semi-Automatic Knowledge Graph Creation and Evaluation

Veronika Hiesinger

Date of Birth: 20.05.2002
Student ID: 12024031

Subject Area: Information Business

Studienkennzahl: J123456789

Supervisors: Stefani Tsaneva and Prof. Dr. Marta Sabou

Date of Submission: 13.10. 2024

Department of Information Systems & Operations Management, Vienna University of Economics and Business, Welthandelsplatz 1, 1020 Vienna, Austria

DEPARTMENT FÜR WIRTSCHAFTSINFORMATIK UND OPERATIONS MANAGEMENT DEPARTMENT OF INFORMATION SYSTEMS AND OPERATIONS MANAGEMENT

Contents

List of Figures

List of Tables

Abstract

Hybrid-Intelligence and the use of Crowdsourcing have become an essential part of Knowledge Graph (KG) creation. There are many different approaches of including human participants in the process of designing a semantic model such as a KG. These approaches have evolved over time, as has the applicability of the finished Graphs. Still, there are some gaps in current research, as a certain lack of understanding about the workings of semi-automatic creation approaches and an incomplete classification of them. In this thesis, we examine methods of KG development by reading through multiple papers concerning this topic and analyzing the described projects. We find that there are distinct ways of grouping such methods, for instance, depending on the means of data extraction or the level of human involvement. Furthermore, different ways of evaluating a Knowledge Graph are explored, as well as possible challenges that need to be overcome when designing a well-working model.

1 Introduction

The creation and evaluation of Knowledge Graphs (KGs) have gained importance and popularity in the last decade, as KGs pose the underlying base of many applications which were designed for facilitating tasks completed solely by humans until now [23]. Hybrid Intelligence is a valuable asset in these creation processes, since it enables developers to employ the strengths of both humans and technology [6]. So far, there have been different approaches on how to divide the work in the most efficient way to acquire a complex yet structured Knowledge Graph [17].

In this thesis, we aim to analyze aforementioned approaches, to facilitate their classification based on differences in the development procedure. We are furthermore interested in the correlation of the cooperation between humans and machine and the field of application the KGs are built for over time. Moreover, we intend to provide an overview of the evaluation methods applicable to such Knowledge Graphs and examine the challenges that are to arise during and after the process. These goals can be composed into the following research questions:

- What are the characteristics of current semi-automatic Knowledge Graph creation approaches and how can they be classified according to their differences?
- How has the cooperation between humans and machine evolved in the past decade?
- How can Knowledge Graphs be evaluated and do developers have to take care of potential challenges and biases?

To answer these questions, we have examined a copious number of papers depicting projects that involve Hybrid-Intelligence in Knowledge Graph creation, the earliest being from 2013 and then gradually moving on to more recent ones. We analyzed these papers on the task division between the system and human workers, on the conducted experiments, how they are evaluated and whether the process contained any complications.

The remainder of this thesis is structured as following: First, we are going to introduce and explain the most important concepts and terminology in Section 2. In Section 3, we describe our methodology. Section 4 outlines the different approaches to Knowledge Graph creation and how we composed them into different categories, to answer the first Research question. Furthermore, it describes the recent trends in applications these Graphs are featured in, as a response to Research Question 2. The last two chapters deal with finding an answer to the last Research Question; in Section 5, we look at methods to evaluate the models, Section 6 deals with challenges and biases in the collaboration of models and human beings.

Lastly, we hope that this thesis provides enough insight into the workings of Knowledge Graph creation for model developers to understand the possibilities they have in their modelling process, and that it might help them to design their projects in the best way possible.

2 Key terms and definitions

This section explains the most important terms that are necessary to understand the different approaches to Knowledge Graph Building.

2.1 Semantic Resources

Hogan et al. [14] define a *Knowledge Graph* as "a graph of data intended to accumulate and convey knowledge of the real world, whose nodes represent entities of interest and whose edges represent potentially different relations between these entities". In other words, a Knowledge Graph consists of entities from one or various real-world topics and the relations between them, all portrayed in a graphical layout with nodes connected by edges. These graphs can be created to help with all kinds of topics, as we will see later on.

A *Knowledge Base (KB)*, as opposed to a Knowledge Graph, is an accumulation of knowledge of a certain topic with entities, descriptions and relations as well, yet it follows a less graphical and more rigid structure. Instead of nodes and edges, a Knowledge Base works in a more table-like manner and focuses on the retrieval of knowledge rather than working out existing relations [28].

2.2 Human-in-the-loop approaches

Crowdsourcing in its simplest form describes the act of outsourcing tasks that would otherwise have to be accomplished by the involved people to volunteers, usually done via the internet [15]. Some of the most popular websites that offer Crowdsourcing are, for example, Amazon Mechanical Turk (MTurk) and CrowdFlower, which are host to a lot of projects aided by volunteer work [11]. In connection to creating and evaluating Knowledge Graphs, Crowdsourcing mostly relies on Human Intelligence Tasks or microtasks.

A *microtask* describes the process of breaking down an extensive job into a considerably large number of smaller tasks that can be done by one person or program alone in a feasible time [21]. If the task requires a human operator to be completed, some tend to use the more concise term *Human Intelligence Task (HIT)* [16]. In this thesis, we are solely interested in microtasks solved by humans, therefore these two terms will be applied synonymously and the word microtasks refers to jobs done by human beings.

2.3 Automated curation approaches

Tools and methods that recur rather often in Knowledge Graph Building are Natural language processing, Information Extraction and Entity Linking. *Natu-*

ral Language Processing (NLP) in this content describes the process of a system or algorithm parsing through texts written in any language that hold semantic meaning and are grammatically correct, in order to process the information in said text [29]. The system is supposed to filter out persons, concepts, objects, etc, the relations between them and should also be able to give explanations for these things. *Information Extraction (IE)* is rather similar to NLP, however, the texts that are processed here don't need to be structured, as IE is possible with tables, lists, or any documents that are not comprised of full grammatical sentences [13]. *Entity Linking (EL)*, following the events of NLP and IE, is responsible for linking the found entities to their already existing nodes, webpages, etc. In case entities have more than one possible node to be linked to, EL tries to find clues in the entities relations as to how to define it [3]. The word "Bulls", for example, can refer to the animal breed as well as to various sports teams, if it is connected to the word "feed" however, it is more likely to relate to the animals.

A *Large Language Model (LLM)* is a program able to generate human language and interact with a human user in the way another human being would do. They understand input in natural language, are able to predict word sequences and produce answers from an underlying pool of text sources. Usually, they are neural networks and use machine learning to expand their "knowledge" without human supervision. The most well-known examples of LLMs are ChatGPT or Gemini [5].

3 Methodology

To answer our Research Questions, we look at a collection of papers connected to semi-automatic Knowledge Graph creation. These papers are taken from the source pool of a systematic literature review conducted by Sabou et al. [41] in 2024 and are, in general, from the 2010s. Additionally, we search for more recent topic-related papers to examine the progress of KG building. After selecting the appropriate projects, we read through them and extract the important information, which is then accumulated in an Excel-sheet and categorized into the elements model name, model use, task division, human tasks, machine tasks, evaluation, training, experiments, cost optimization and challenges - as can be seen in the appendix.

Once the data extraction is completed, we group the outcomes in the spreadsheet and search for possible categorization methods: every paper's characteristics are highlighted and those with matching characteristics are accumulated into a category. Of course, one paper can be a part of multiple categories, which we can see in the next section. For the third Research Question, we filter for common evaluation methods and challenges. Lastly, we pick out one piece of literature that deals exclusively with biases in Crowdsourcing to draw conclusions on that area.

The papers used for the analysis are comprised in the following table:

Paper reference in Excel sheet/ Reference Number	Name	Year
1/[10]	Large-scale linked data integration using probabilistic reasoning and crowdsourcing	2013
2/[39]	The BBC World Service Archive Prototype	2014
3/[17]	Combining information extraction and human computing for crowdsourced knowledge acquisition	2014
4/[33]	Exploiting users' feedbacks: Towards a task-based evaluation of application ontologies throughout their lifecycle	2015
5/[6]	KATARA: A Data Cleaning System Powered by Knowledge Bases and Crowdsourcing.	2015
6/[22]	Refining Automatically Extracted Knowledge Bases Using Crowdsourcing	2017
7/[24]	Use of Ontology Structure and Bayesian Models to Aid the Crowdsourcing of ICD-11 Sanctioning Rules	2017
8/[31]	Kgeval: Accuracy estimation of automatically constructed knowledge graphs	2017
9/[1]	Detecting Linked Data quality issues via crowdsourcing: A DBpedia study	2018
10/[23]	OC-2-KB: integrating crowdsourcing into an obesity and cancer knowledge base curation system	2018
11/[12]	Efficient Knowledge Graph Accuracy Evaluation	2019
12/[34]	You are Missing a Concept! Enhancing Ontology-Based Data Access with Evolving Ontologies	2019
13/[36]	Evaluating Knowledge Graph Accuracy Powered by Optimized Human-machine Collaboration	2019
A/[38]	An optimized task assignment framework based on crowdsourcing knowledge graph and prediction	2023
B/[30]	Creating and validating a scholarly knowledge graph using natural language processing and microtask crowdsourcing	2024
C/[25]	Achieving Knowledge-as-a-Service in IIoT-driven smart manufacturing: A crowdsourcing-based continuous enrichment method for Industrial Knowledge Graph	2022
D/[18]	Hc-covid: A hierarchical crowdsource knowledge graph approach to explainable covid-19 misinformation detection	2022
E/[19]	CrowdGraph: A crowdsourcing multi-modal knowledge graph approach to explainable fauxtography detection	2022
F/[9]	Implicit bias in crowdsourced knowledge graphs	2019

Table 1: List of papers examined in this thesis

4 Semi-Automatic Knowledge Graph Creation Approaches

When it comes to classifying different types of building techniques for Knowledge Graphs, there are various ways to divide the methods into groups. In order to gain these insights, we conducted a thorough analysis of multiple papers depicting building a Knowledge Graph with the help of Crowdsourcing, filtered for differences in their processes and analyzed the involvement and workload splitting between human workers and machines. To see whether the levels of collaboration have changed since Crowdsourcing in connection to Knowledge Graphs was introduced in the early 2010s, we looked at both older and more recent projects.

The following sections will introduce and explain methods to categorize Knowledge Graph Creation techniques and give an overview about how these techniques have developed in the last decade.

4.1 Parsing of underlying Knowledge Bases

In most cases the creation of Knowledge Graphs follows the system of first creating the nodes and links, then looking for faulty or incorrect relations that need to be further looked into and having workers examine these links. However, all three of these steps can be accomplished in various ways.

For the first step of creating a primary Knowledge Graph, the information about the topic at hand needs to be extracted from existing sources, as for example, textual documents, social media posts, reports or already existing Knowledge Bases [6]. This can either be done manually by people or by parsing these sources with an NLP program.

Table 2 sums up which projects used an automatic or manual approach.

Automatic Approach A majority of the papers studied, like [17] and [1] showcased the use of NLP programs like CSO classifier. In this case, up to thousands of documents are read by the systems in order to find the most important entities and link them together with semantically correct relations, which involves a lot of IE and EL. All of the created nodes and edges are combined into a first version of the desired Knowledge Graph.

An example of this process is described in [17]. The aim of this paper is to create a Knowledge Graph for popular pieces of Literature, like "Harry Potter" or "The Lord of the Rings", where users can search for characters, places or events and get an overview about how the chosen entity is related to others. To achieve this, an NLP program filters through datasets describing the books, or the books themselves, and creates triples depicting the information gathered. The triples don't just refer to the fictional characters and relations, but to metadata as well, for example, by whom and when the book was written. One important feature the paper's model has, is symmetry in the triples. This

KG Creation Approach	Papers employing this approach
Automatic	[10], [39], [17], [6], [22], [31], [1], [23], [30], [25]
Manual	[33], [24], [18], [19]

Table 2: Automatic and Manual Creation Approaches

means, that if, for instance, the triple "Animal farm" "written by" "George Orwell" exists in the Knowledge Graph, the triple "George Orwell" "wrote" "Animal farm" is automatically constructed as well. Symmetry helps decreasing the runtime when it comes to searching a specific connection in a Knowledge Graph, as the edges can be accessed from two sides. It furthermore helps in reducing the number of triples chosen for crowdsourcing, as we will see later in this section.

The model from [39] works on the same premise but takes the complexity of the Information Extraction one step further even. The project aims to create an archive of the BBC's Audio and Video programs, and annotates them with helpful information for the viewer, in order to get a better overview of what the show is about. Just as with the model described above, it is supposed to provide both data, like the topic, the story and the plot of the program, and metadata, like the host, narrator, release date or actors contributing to the display. Contrary to other cases of IE and EL, it not only filters through textual documents, but also through Audio and Video files. On the one hand, the spoken sentences are transcribed in order to add on to the entities and relations already found in the articles, posts and reports. On the other hand, the voices and faces are compared, classified and matched to the characters, reporters and real-life people from the Knowledge Graph. This helps to further expand the information provided, and moreover enables the model to make connections between programs that feature the same people or voices, either as narrators or participants.

Manual Approach Contrary to projects where the foundational Knowledge Graph is constructed by an NLP program, there are also those models who use the intelligence of human-beings to construct their baseline, like [24]. In this case, it is the human's responsibility to decide which types of information are the most relevant to be included in the graph, and how the relations should be handled. The level of activity the experts need to exert in this task can vary; for some projects, workers created the entire Knowledge Graph foundation, which, of course, is then very limited in size. In other projects, workers found a different approach, for example to only choose the topics and entities the Graph should be able to provide information for, and then leaving the learning process to the model and the Crowdsourcing.

An example for the later type of process is shown in [24]. The aim of this model is to create a Knowledge Graph that helps with the international classification of diseases. Users should be able to ask the model a question regarding a medical disease and get an answer with the help of the Graph. Before the

research for entities and relations is even started by the program, experts on the topic put together a pool of questions the model should answer once it is complete. These questions are typically yes/no questions designed to facilitate the easy handling of finding out about the user's illnesses. Once these questions are created, the model searches for the most systematic way of forwarding these questions to the crowdworkers, and only once they are done, the information is incorporated into the Knowledge Graph via nodes and edges.

Another way of creating the underlying Knowledge Graph by humans has become more and more popular in the recent years, especially when it comes to models that help navigate and find incorrect statements on social media. Both [19] and [18] are projects designed to automatically recognize incorrect Twitter posts and point them out as such. The model from [18] on one side aims to filter out wrong information about Covid-19. The first step of the creation process is to have a group of both experts and non-experts read through factually correct articles about the triggers, average duration and similar truths about the Covid disease, and arrange all these facts into triples that form a baseline Knowledge Graph. What is special about this project, is the fact that first, only the non-expert participants work on the Graph, and only when they are done, the experts on the field extend that Graph with their professional inputs. The reason to arrange the building in this way is to create a two-step hierarchy of the terms, which means a lot of entities are synonyms of each other, made possible by the use of both professional and unprofessional terms. This way, the filtering for answers in the later use stage of the Knowledge Graph is a lot faster than when every entity exists exactly once.

The building process of [19], on the other hand, doesn't use a two-step hierarchy; it does, however, aim to supply users with a graph able to work on multi-modal Twitter posts. This model concentrates on more general information than Covid, mostly looking for information about worldly-known celebrities and people with significant power, as for example, the president of the United States. A notable feature of this system is its multi-modality, meaning the model not only finds out incorrect facts about textual posts, but can also examine the correctness of a picture or photograph. In order to achieve this, workers sort through Twitter posts that include visual components as well. The model learns to link an entity to a face and is then later able to decide whether the picture in the Tweet fits the person described by the caption. Once the workers have composed enough triples, a foundational Knowledge Graph is devised, and through tests run with Crowdsourcing, questionable triples can be either falsified or verified.

4.2 Selection of links incorporated in Crowdsourcing

Once the underlying Knowledge Graph for the model is created, the right number of links fit for microtasks needs to be selected. In some cases, the entire Graph can be annotated, in other cases, the system selects those links it is unsure of and wants to have verified.

Type of Annotation	Papers employing this approach
Complete	[39], [17], [33], [1], [34], [25], [38]
Partial	[10], [6], [22], [24], [31], [23], [12], [36], [30]

Table 3: Complete and Partial Annotation

Table 3 shows the models that can fully be worked on by crowdworkers and those where only selected links can be annotated.

Complete Annotation One example for giving crowdworkers the chance to examine every single belief is the already described archive of BBC's programmes [39]. After the completion of the foundational Graph, users are welcomed to select a certain show and examine the correctness of the entities linked to the episode. If, for instance, the topic of a nature documentary fits what is shown in the Video, a user can upvote the link. Contrary, if the narrator seems to have been identified incorrectly, workers can downvote or report that connection. Users can access every video or podcast on the archive, meaning the Knowledge Graph allows every single link to be examined and worked upon.

The authors of [34] describe another case where every link can be investigated. In this project, multiple ontologies to various topics are accessible to workers, who browse through the triples and analyze them according to their semantic correctness, and the right classification of the entities. If any of the links are faulty, the volunteers are able to look for replacement triples; if the majority of the operatives vote to change the relation, the system adjusts the triple to the humans' recommendations.

Partial Annotation Opposed to being able to work on all links, there are also many models, that strictly select which of the triples get to the Crowdsourcing stage. In this case, the selection process can look very differently:

The model from [6] is responsible for cleaning existing noisy Knowledge Graphs or Bases. It scans through the entire data set and searches for triples or tuples that appear often enough to possibly portray a semantically meaningful relation. It then calculates the probability for this links to be correct, taking into account multiple factors such as frequency, number of other links from the same entity or entity classification. Once this is done, the system selects those tuples and triples with the lowest confidence score and presents them to workers to verify or repair the relations. What is special about this model, however, is the fact that it already suggests repair options for those connections it considers faulty; meaning, before letting humans work on the triples and tuples, it looks for alternative links itself, and then asks which of these alternatives people consider the best fitting. Nonetheless, in order for this process to work, the confidence level calculations need to be immaculate, as the model computes the alternative links with the help of those connections it is certain of.

A different way to select the relations for the Crowdsourcing step is presented in [24]. After the in Section 3.1 described process of making experts put together a question set for the international classification of diseases, the model looks through the question catalogue and calculates the most systematic way to select the queries in order to achieve as much insight with as little HITs as possible. This means, it singles out questions posed more than one time, and erases repetitiveness and redundancy. While the workers try to solve the tasks, the model itself already looks for answers to the questions, creates links and is therefore able to create a new selection of tasks for the next Crowdsourcing batch.

A last example for an unusually designed selection process poses the project described in [36]. It is aimed to discover how to reduce costs in Knowledge Graph creation featuring Crowdsourcing. The chosen way of discovering that was to firstly select a random sample batch from an existing Knowledge Graph, with no real system behind it. Once this batch was sent to the workers and the first annotations were sent back, however,the model started to systematically select the next batches after what had been marked as incorrect in the first sample. The more triples and links are looked at by the workers, the more concise the model can be when strategically filtering for the connections that necessarily have to be looked at. Though this method didn't turn out to be very effective timewise, since the batches have to be sent back and forth more often than in other approaches, it significantly reduced the number of triples that were developed into HITs unnecessarily and decreased the level of repetition in the microtasks.

4.3 Virtual Spaces for Crowdsourcing

Another critical way of distinguishing between different methods in Crowdsourcing is the space the workers solve their tasks at. Developers can either choose a platform specifically designed for microtasks, or the place for solving the HITs is the website of the application itself.

Table 4 summarises which spaces are used for Crowdsourcing by which projects.

Direct Crowdsourcing The BBC archive already described is a prime example for the second category, where users interact directly with the implementation. Another project that applies this method is outlined in [38]. This model is different to many others, because it aims to improve the distribution of microtasks in case the geographical location of the workers is of essence. This process is helpful in use cases like a Taxi agency, where it is usually desired to have cars equally distributed in the covered area, in order to let customers wait as little as possible. The model is developed by using an underlying Knowledge Graph about the workers' locations and their average time to fulfill the assigned task. All the participants have to do then, is interact with the Graph; meaning they accept the charge, travel to the designated place, carry it out and

Crowdsourcing Approach	Papers employing this approach
Direct	[39], [33], [12], [34], [25], [38]
Indirect	[10], [17], [6], [22], [31], [24], [1], [23], [30], [19]

Table 4: Direct and Indirect Crowdsourcing Approaches

then give feedback about how long it took them and how well the route was planned. Afterwards, the model compares all the workers, and can improve the Graph by singling out the ideal person for every location. In order for this to work, of course, the participants need to work together with the application and different locations themselves, and not solve tasks on a website which neglects geographical facts.

In many other cases, like [12] or [34], crowdworkers help enhance already existing Knowledge Graphs by adding new entities or relations to the foundational Graph, to make sure the information always stays updated. This is usually done directly on the application the Graph was designed for, instead of pages like Amazon MTurk.

Indirect Crowdsourcing via marketplaces However, whenever workers are supposed to solve tasks about the triples the model needs help with, rather than simply interacting with the Graph, MTurk, CrowdFlower and similar Crowdsourcing marketplaces come into use.

The authors of [1] depicts a typical use of microtasks. An underlying Knowledge Graph was built by using NLP, and once this is done, the work is outsourced to Amazon MTurk. In a first step, a competition is held for a group of expert workers to find erroneous triples in the Graph, with the person identifying the most mistakes receiving an award - usually money. Once an appropriate number of faulty relations has been determined, those links are processed into HITs; now experts and non-experts alike have to verify that the triple is indeed incorrect, and find suitable corrections for those errors, for example suggesting the right entity classifications or relation descriptions. The microtasks consist of True/False questions for the verification part and multiple-choice questions for the improvement part.

Another case of utilizing Crowdsourcing marketplaces is described in [22]. Existing Knowledge Bases are taken from NELL with the aim to reduce noisy relations, and compromised into a relational Graph. The model sorts out those edges it is unsure about and processes them into microtasks to verify their semantic correctness. Workers sort through these links and color them according to their confidence level of the semantic truthfulness; green for a high confidence that the relation is right, red for being certain the link is meaningless, and blue for not knowing the triple's components. After workers have completed their tasks, the model automatically colors all the edges related to a colored link, in this paper called sub-vertices, in the same color. Through this, it is possible to verify a lot of synonymous edges by just letting one of them be worked on, and the other way around as well; if a link gets colored red, all the links with similar

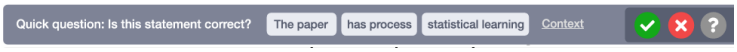

Figure 1: A Yes/No question microtask, [30]

meanings have a high probability of being incorrect, too. Those entities with a blue coloring might imply that something went wrong with the classification or description, consequently suggesting that other edges from that same entity could also not apply in the correct way.

4.4 Formats of Microtasks

There are many different ways to create a microtask in a Crowdsourcing marketplace. Although a few have already been introduced in the previous sections, an overview will be given underneath:

A very popular and frequently used option are *Multiple-choice questions*. Here, workers are supplied with triples or relations where one of the components is missing. Usually, they are given between three and five options to choose the missing part from, and the majority of the voters decides which term fills the blank space best, in a semantic way. It is also possible to let workers decide which is the best classification option for an entity; in this case, the triple is already complete, but one of the entities needs a description. The outcome is decided by majority vote as well. This approach is used in [17].

A similar approach are *True or False questions*, sometimes also *Yes or No questions*. Workers aren't given a triple with a missing component, but a relation that is already complete. Their task is to decide whether both the semantic meaning of the link and the entity classification are correct. Usually, they are also given the option to choose "I am not sure about the correctness." An example is depicted in Figure 1, taken from [30]. True or False questions are furthermore applied in [6] and [24].

Alike to Multiple-choice there are the *Open-gap questions*. They work after the same principle, the only difference is that there are no options to choose from, but workers have to come up with their own relations, descriptions and classifications for the blank spaces. This method is employed in [17].

A concept that works a little differently is the *Upvoting or Downvoting of connections* according to relevance. This method is used by the BBC archive. Here, workers look at a picture, article, video, or any kind of content, and sort through recommended entities like topics or people that might be related to the content. The model suggests an order for these entities, sorted from most relevant to the file to least relevant; if, in the workers' opinions, that order of relevancy is incorrect, they can vote to move certain concepts up or down. This is used in [39].

Coloring of vertices has already been described. In this method, workers look at relations and links between entities, and color them according to whether the edges are semantically correct or not. It is applied in [22].

Figure 2: Annotating and providing background information about Barack Obama, with the obligation to name the sources

Usually, the model itself is responsible for finding the triples and relations it is uncertain of. Sometimes, however, that process is outsourced to crowdworkers and they *look for erroneous links*. In most cases, that is a job for experts in the respective topic, something that needs to be mentioned on a Crowdsourcing marketplace in order to avoid unprofessional opinions creating noise. Sometimes, this process is followed by a second step, where the incorrect links are given to non-experts, who try to validate the decision made. *Validation of correct or incorrect triples* is a rather popular microtask, as in many cases, it can be combined with creating symmetric relations, meaning both ways of an edge can be verified or falsified in just one task. These concepts are employed in [31].

A task that requires a bit more active involvement by the worker is *annotating triples or entities*. It has similarities to open-gap questions, but usually the humans have to do a bit more research for it. Mostly, the task is meant to accumulate background information to entities, or to provide metadata for the collected information, as for instance, weblinks to famous people. An example is provided in Figure 2; a project that used this method is [36].

Another method where the users have to engage themselves quite a lot is *devising questions*. Here, they are given a short paragraph about a certain topic related to the underlying Knowledge Graph, with the instruction to think of a question that is answered by the content of the paragraph. Once the question is posed, the model works out an answer to the question with the help of its Graph and sends that answer back to the worker. If the human doesn't accept the answer as satisfying, the model knows that the Knowledge Graph is not yet at the level of conveying the Information of the underlying textual documents and looks for ways to improve the triples. In other cases, the human is asked to come up with an answer themselves. This answer is then compared to how the model would have replied, and if there are not enough similarities between these two answers, the Graph has to be reevaluated. Figure 3 shows how this process can look. This method is used in [25].

The task that assumably takes the most effort from workers is having to read through articles, documents or social media posts, and then create triples

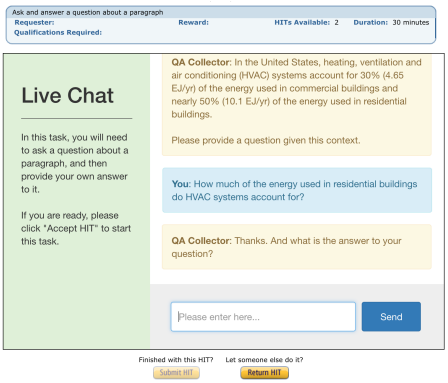

Figure 3: The worker is asked to create a question about the content of the given paragraph

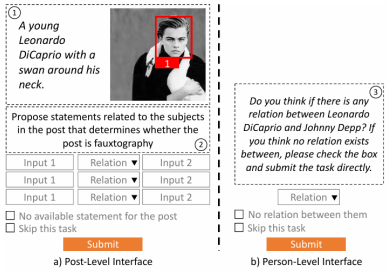

Figure 4: Creating triples from a text and related image, [19]

on their own, either with or without the help of a user assistant. In some cases, like [34], assistance is provided by the model to find the right vocabulary to describe the relations from the source material; this way, it is easier to find classifications and keep the Graph as accessible as possible. On other occasions, as mentioned in the projects from [19] and [18], which aim to recognize incorrect posts on social media - already described in section 3.1 - workers have to create links completely on their own. On the Crowdsourcing marketplace, they receive posts and pictures, and then have to extract information and compose this data into triples. Figure 4 shows how this works,taken from [19]. Other projects where crowdworkers created triples are [18] and [34].

4.5 Level of Human Activity

As we have seen in the different methodologies of Crowdsourcing, there are various levels of human activity. In some cases, workers only need to interact with the model, without having to come up with any input themselves. In other cases, the underlying Knowledge Graph is created by the model itself with the

Creation Method	Papers employing this method
Method 1	[39], [33], [24], [1], [23], [25], [38]
Method 2	[10], [17], [5], [22], [31], [12], [36], [30]
Method 3	[34], [18], [19]

Table 5: Distribution of Creation Methods used

help of an NLP program, so that humans only have to apply their knowledge to look for incorrect links. Then there are the cases where workers create the Knowledge Graph themselves, meaning they have to take care of the hardest part of the process. Of course, there are also variations of these three methods of how to design a Knowledge Graph that lie somewhere in between. This section, however, will focus on the three methodologies described above and portray them with graphical aids.

Table 5 provides a summary of the three methods used in the studied papers, which will be explained in the following pages.

Figure 5 shows the process of mainly including crowdworkers for interacting purposes, deployed, for example, in [39]. These kinds of projects are started by having a Natural Language processing program read through a considerably big amount of textual, visual or audio documents, and extract data like people, places, dates, groups or concepts in general from those. Once this is done, the semantic definition of this data is generated from the given contexts, and the concepts are composed into entities with classifications. In the following step, all the entities are linked together with relation tags, and this way, a Knowledge Graph with nodes and edges is created. This Graph is then embedded into the project's application and published, either for test runs or for general use. Users are invited to interact with the Graph, either through asking the model questions about the topic at hand, through navigating archives via related themes or through following the instructions the model supplies them with. We have already experienced this kind of interaction with the BBC archives or the Taxi Driver Knowledge Graph for applications with spatial and geographical relevance.

While the users interact with the application, the model collects information about these transactions. Usually, users get asked directly whether they are satisfied with their experience with the system, or they have the chance to leave annotations on the answers they received from the Knowledge Graph. Additionally, many models are able to analyze how pleased the workers were through the runtime of people's queries; if, for instance, a question gets answered by the system the first time, but is reformulated by the user to obtain a different reply, the algorithm understands that the first answer was unhelpful in the users opinion. The faster a user finishes a query or finds the content he was looking for, the better organized and easier to navigate the Knowledge Graph; from this, the system can learn how to structure the triples.

If the model calculates that users are satisfied with their application expe-

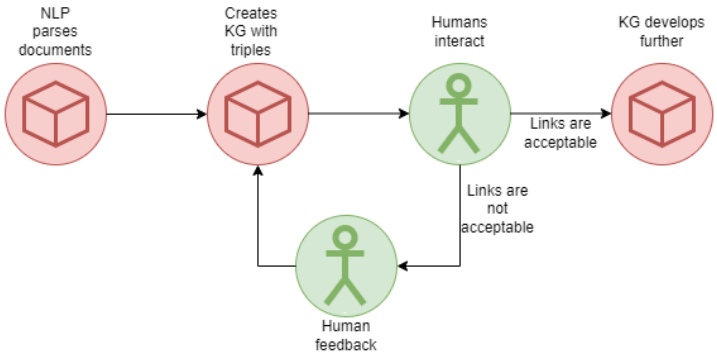

Figure 5: Method 1: Crowdsourcing through interaction

rience, either from analyzing the length of interaction or asking directly, the triples which were used for the worker's query stay, and with them the Knowledge Graph is extended further. This means, that the model can be confident about the relations' correctness and use them as a foundation for new edges in the Graph. However, if the triples do not seem to be trustworthy, a feedback loop is started; this loop can be done through asking users what they had expected from the answers or the process, and integrate these notions into the triples. In some cases, asking the users can be skipped, and the system immediately changes the triples without consulting human opinions first. In both cases, the adjustments are incorporated into the application, and the user interactions continue. This loop lasts as long as it is necessary to create a graph without any faulty relations.

Advantages of this method are that it enables the project to draw from a bigger database than if humans were responsible for creating the triples. Furthermore, the creation process doesn't take too long, and the application can be started relatively fast. The downside to this, however, is that the system is endangered of running with a faulty version, and the users' requests can't be answered correctly until the Graph is truly finished. This trade-off of speed and factual accuracy is something to be considered when deciding a Crowdsourcing method for a project. There is no requirement for the users to be experts in the respective field, they should just possess enough intelligence on the topic to make productive contributions. To eliminate distortions due to unprofessional inputs, a majority vote is advisable wherever possible.

In Figure 6, we can see the method of using crowdworkers to verify correct or incorrect triples. The beginning of this process is equal to the one shown above, with an NLP filtering through textual documents and finding entities and the relations between them, which are then composed into triples in the Knowledge Graph. Once this is accomplished, the model creates confidence scores of all the

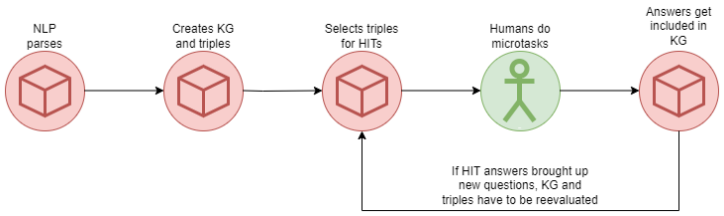

Figure 6: Method 2: Crowdsourcing by operating on selected relations

triples and then selects those links that are fit to be worked on in microtasks. The selection process can either focus on verifying those relations the system is confident in, or finding errors in those with a low confidence level. The election of suitable triples can have all sorts of additional features, like the included recommendation of alternative triples, as described in section 3.2. What the majority of the projects have in common, however, is to single out certain links. This poses a major difference to having workers operate on the application itself, where they have access to every single triple - or at least all currently available triples - in the Knowledge Graph.

Now it is time for the system - either the model itself or the model's developers - to create microtasks. As shown above, there are many different ways to compose microtasks, and developers have to decide which method is best fitting for the goal they want to accomplish through the Crowdsourcing. True or False questions are usually a very reliable way of verifying triples the model is already sure of, yet sometimes a multiple-choice task might help with gaining a bigger picture, for example when classifying an entity. Whatever choice is taken, afterwards the HITs are forwarded to the workers, and the system needs to wait until all, or at least some, of the tasks are solved.

After an adequate period of time, the answers are analyzed by the model and integrated into the Knowledge Graph. Links that are verified can be strengthened to create other triples with, and those which were proven to be incorrect can be rectified. In case that some of the answers did not fall out as expected by the model, it needs to go back to the step of selecting triples for the microtasks. The HITs enable the system to find out where the errors occur, and this can only be accomplished by trying out as many links as possible until the source of the mistake is found. The consequence can be a very lengthy loop of solving microtasks, adjusting the correct triples and singling out the incorrect ones, yet it is a reliable way of finding the roots of fallacy.

The advantages this method brings is that, as in the case above, it enables the model to have a great number of baseline information and a very intricate and complex graph, due to an algorithm creating the Knowledge Graph. Furthermore, it allows the developers to use all kinds of HIT methodologies, which of course, creates the possibility to come up with the best fitted microtasks for the Crowdsourcing. A downside, however, is that the lengthy feedback loop can prevent the application from being released for a considerably long time,

which might be a problem, even if the finished version has a high probability to be wholly correct. Another disadvantage is the fact that the microtasks are usually executed on Crowdsourcing marketplaces, where payment for the workers is obligatory. This, on the other hand means, that most of the people are competent enough to not distort the results; that is a trade-off the developer needs to consider. Depending on the topic, experts might be required, which can easily be found on webpages like MTurk or CrowdFlower. This method is used in [30], for example.

Last, Figure 7 depicts the process with the most human activity, applied in the project of [18]. The process is started by human workers reading through textual or multi-modal documents, as we have seen with the models who aim to recognize incorrect social media posts, for example. Humans perform the task that in the other two methods was carried out by a Natural Language Processing program: extracting entities from the texts, classifying them as key concepts and linking them together with semantically meaningful relation tags. In some cases, they may be aided by an assistant algorithm, which helps them in finding the correct wordings, suggests tags or classes and overall assists in keeping the triples complex, yet not confusing. In other cases, the workers are truly on their own, and have to think of a structured way to compose their relations. On these terms, it is advisable to let people with experience in creating Knowledge Graphs work out the triples.

Once all the data is collected from the sources and the triples are finished, the model accumulates them into a Knowledge Graph. Now, it is time for a test run: usually, private experiments are conducted to examine and evaluate the work of the model. In these experiments, the system runs tests in its field of later application, whether that may be answering questions to a certain topic, assisting in research or finding fake information publicly available in the web. Either the developers, crowdworkers or both look through the results of the test run, and compare the model's answers to the gold standard outcome they know to be correct. Once again, there are two possible steps to take afterwards; if the model's results turn out acceptable, the triples are kept in the Knowledge Graph, since they seem to help in achieving the intended aim of the application. If this is not the case, the Graph needs to be reevaluated and restructured. This means, that the system has to look through the triples and search for possible alternative connections. Should this not be effective enough, the process goes even further back, to the point where humans create new triples which are better fitting for the model. This is done until the application works flawlessly, and even then, a constant feedback from human workers is recommendable.

A given disadvantage of this method is that humans take much longer to read through texts, watch videos or listen to audio files: due to this, the underlying databases are usually much smaller than when an NLP parser creates the triples, and it takes a longer time as well. One thing, however, that distinguishes humans from machines in a very positive matter, is their ability to pick up on social cues, read between the lines and understand unspoken statements, something that algorithms do not possess. An example for this is stated in [19],

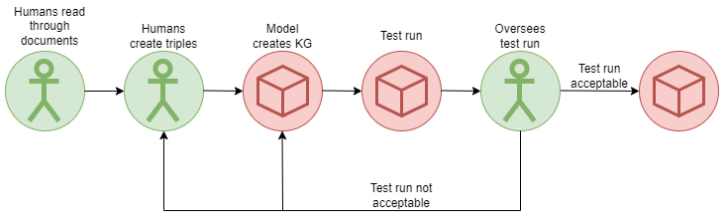

Figure 7: Method 3: Creating the triples of the Knowledge Graph through Crowdsourcing

the project that developed a Knowledge Graph able to find incorrect Twitter posts: most people know that Donald Trump isn't very fond of Joe Biden; yet, for a model, this fact is hard to understand. Since the wording of political debates and Social Media posts is still, at least to a certain degree, professional, and the dislike between people is transported through gestures, voice tones or facial expressions, it can be difficult for a machine to understand inter-human relationships. Because of this, humans need to set the foundation of how to interpret these associations, especially for applications that are aimed for being confronted with human emotions.

This of course means, that the developers need to hire people who know how to read social clues. Distortions by workers who interpret certain actions differently than others can be avoided by only integrating triples into the Graph that have been suggested by more than one person, or at least hold the same meaning as other created links.

4.6 Recent Trends

One conclusion that was drawn from researching and analyzing both recent and older projects, is that both the fields of application and the Crowdsourcing methodologies have changed. The following section will explore these changes.

Models from the early to mid-2010s mainly had the purpose of creating Knowledge Graphs just for the sake of collecting intelligence regarding a certain topic and capturing it all in one place. Many papers don't mention any further use cases for these Knowledge Graphs, the main aim was to create a Graph with the help of crowdworkers and perfect it. This can be seen in papers like [17], which had the goal of creating a Knowledge Graph about literature; this was certainly an interesting process, yet the authors did not name any further use for such a graph. Similar is [10], a project that tested out Information Extraction and Entity Linking on various topics, but did not seem to have any following plans for these Knowledge Graph either.

Another popular process in this earlier time period was improving already existing Knowledge Graphs. The project from [6], for example, describes the procedure of creating a model that filters out triples from Graphs that are likely

to be incorrect, and creating alternatives for these faulty relations. Although the focus of the project clearly was to perfect the alternative-recommendation, there is still no real major goal behind it. Another case is [22], which had the aim of reducing noise in existing Knowledge Bases and improving data extraction.

There are only a few more exploratory cases from this decade, which had a more applied goal in mind, and those were developed in its later half. For instance, there is the BBC archive, that has already been described in previous sections, to help users navigate easily through the Broadcasting company's streaming websites. Other cases are [23] and [24]; the first one was developed to help link cases of obesity to the chances of surviving cancer, an intention that supports doctors with prognosticating patients and finding the optimal treatment measures for individual cases. The second one was intended for a medical field as well, namely obtaining rules for the identification of internationally known diseases. Both of these projects clearly had the aim to facilitate an aspect of day-to-day life and were released rather late in the 2010s, which shows the trajectory of Hybrid Intelligence in Knowledge Graph Creation, from solely composing Graphs to finding real-life applications for them.

Most of these projects from the 2010s used the second method from Section 3.4, where humans solved microtasks about the Knowledge Graph. A few models used interaction with the workers as well, but people were not yet included in the creation process of the Graph.

From 2020 on, however, projects became a bit more hands-on. Knowledge Graphs were created to support an application used in daily life, and those also migrated in the direction of systems for everybody, not just researchers and academical users.

Many projects showcase this shift, as for example [38], which has the aim of improving the geographical distribution skills for tasks of a model, to help Taxi-businesses create the perfect underlying Knowledge Graph for their workers, for instance. Another case is described in [25]: this model was developed for apprentices learning their craft. They are supplied with an assistant which is able to answer every question relevant to their occupation: it can, for instance, give a step-by-step instruction of how to assemble a fridge. The triples for the baseline Graph are composed by experts of the respective fields. But Knowledge Graphs created by Crowdsourcing don't have to be limited to labor environments. The two applications that help recognizing fake posts on Twitter, either Covid-19 related in [18] or Tweets about celebrities in [19], especially demonstrate that nowadays these projects are created for the use of everybody - at least everybody with an internet access.

The majority of these models are developed by having humans create the triples for the Knowledge Graphs. Some applications are enhanced by user interaction, yet, having workers do microtasks about a Graph composed with the help of Natural Language Processing is not as popular and frequent as it was a decade ago.

5 Evaluation methods

There are many different ways to measure and evaluate the on-goings of a system operating on a foundational Knowledge Graph. The following section explains some of those methods.

Prediction Accuracy Benchmarks *Precision* is defined by the percentage of Triples correctly labeled as correct in the baseline Knowledge Graph. The measure is a typical benchmark for evaluating entity classification. The metric is calculated by looking at the triples the model labels as correct. The number of triples that indeed were correct (True Positives TP) is divided by the overall number of links predicted as correct, including those that turned out to be false (False Positives FP):

$$Precision = \frac{TP}{TP + FP}$$

In other words, precision answers the question of how many of the triples labeled correct by the model actually were correct. The closer this number is to 1, the more reliable is the system in predicting the correctness of a triple. This metric is used, for example, in [10].

Recall usually goes hand in hand with precision, as it is the percentage of correctly identified correct triples. It is calculated by dividing the number of correct triples labeled as correct (TP) by the number of all overall correct triples, including those that the model did not predict to be right (False Negatives FP):

$$Recall = \frac{TP}{TP + FN}$$

Recall answers how many of the correct triples the model identified. The closer this number is to 1, the better the model is in finding all correct triples.

Precision and Recall are often used as a combined benchmark, but there is a certain trade-off to it; if a model is focused on finding all positives, it may also include a few incorrect triples. If its aim is to only select correct links, chances that a few do not get elected are very high. Developers need to decide which of those two options has a higher importance, and test their model accordingly.

This trade-off, however, can be managed by calculating the *F1-score*, which captures the balance between Precision and Recall. It is calculated in the following way

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall}$$

and ranges between 0 and 1; 1 indicates a perfect balance between the two metrics, and 0 means that one of them is significantly better than the other.

This evaluation method is widely used with models that have a foundational Knowledge Graph created by Natural Language Processing, where crowdworkers check on the correctness of the triples. It is easy to calculate and furthermore very straightforward in its interpretation. A Project that used this metric was, for instance, [17].

One more, less customary benchmark that works with Negatives and Positives is the *Specificity*. It measures a model's ability to exclude incorrect triples. It is calculated by dividing the untruthful links identified as incorrect (True Negatives NP) with the number of all false triples, including those that were unlawfully labeled as correct (FP):

$$Specificity = \frac{TN}{TN + FP}$$

Specificity is not used as frequently as precision and recall, as it is often difficult to really measure the number of true negatives. In some cases, experts aim to include even those triples from the base material the model didn't even create, or created but discarded, which of course enlarges the factors significantly. However, in case there is a meaningful agreement on how to define the group, it serves as a useful benchmark on how well the project manages to identify and exclude incorrect links, as for example in [24].

One further metric that is often used to evaluate the operation methods of a Knowledge Graph is to compare the outcomes to a so-called *Gold Standard*. This is mainly applied in cases where the model is supposed to find correct triples from an underlying text. To test the workings, experts on the topics create the correct links from those sentences that are to be evaluated; these triples serve as the 'golden rule' and are assumed to be truthful. Once the model has created relations itself, they are compared to these expert-made orientation triples. Depending on the differences between the template and the solutions, the *accuracy* is calculated in the following way:

$$Accuracy = \frac{number\,of\,correctly\,created\,instances}{number\,of\,total\,created\,instances\,in\,test\,pool}$$

In general, accuracy is used to calculate the percentage of how much was created correctly by the model when compared to the gold standard; due to the simple way of calculation, it can be applied to all areas that might need evaluation. The accuracy of triples, entities, relations, classifications or even the prediction of links - if not every necessary component is given - can be generated this way. Usually, accuracy is paired with precision and recall or, in case the data is very imbalanced and there are many more instances of one type than of the others, weighted accuracy can be applied, where the distribution of certain factors is considered and taken into account in the calculations. The model from [31], for instance, is evaluated with this method.

Comparison by Confidence Benchmarks Another way of assessing a Knowledge Graph's effectiveness is the evaluation of the *confidence level*, a topic which has already been shortly breached in the above sections.

The confidence level is a tool widely used in the whole life cycle of Graph creation. It can be calculated by the model itself as well as humans. In the stages of the development process, it often occurs that the system creates confidence

scores for its triples in order to decide which will be outsourced to crowdworkers; after the Graph has been completed, the certainty levels for each triple of the end product are assessed by experts. This is practical when comparing Knowledge Graphs with similar subjects, and furthermore serves as an optimal way of finding a starting point for improvement.

As explained by Paulheim in [32], the confidence score is a number ranging between 0 and 1 assigned to each link or classification in a Knowledge Graph. This confidence can be aggregated through different reasons; not everything from the underlying source material has to be trustworthy, and triples created from documents with lacking integrity usually receive a lower rating. In other cases, the model might not fully understand the meaning of some sentences and, being aware of this, puts a lower value on the links stemming from those. Often, machines phrase connections differently than humans would, which leads to workers rewarding these links with decreasing scores, as such wordings can lead to confusion or semantic distortions.

At the end of a Knowledge Graph creation process, the confidence score plays two major roles: on one hand, the average or weighted average level of certainty is calculated, to examine how well the Graph is doing in general and to see whether it has to be revised as a whole. On the other hand, it is customary to set a confidence threshold - a fixed scale, for example 0,8. All triples and classifications below this number are either cut out of the graph or sent back to crowdworkers to restart the building process. The threshold method is a reliable way to further find out whether the confidence is imbalanced, for example, if the model is extraordinarily unsure of entities of a specific type. This helps to set the focus on where to improve and work in different ways than before.

Another feature confidence scores entail is to enhance link prediction. In case a triple is incomplete, the model can supply alternative recommendations for the missing piece and rank them according to how confident it is in each of the respective options. The more often the top alternatives are picked by crowdworkers, the better a model can improve its predictions.

All in all, the confidence score is an exceptional way for a system to evaluate itself. Different to other measures, it doesn't need human input to check on correctness or compare with a truthful outline, but is able to find weak links on its own. Still, human participation and occasional intervention is advisable to assess the ongoings in the graph, and decide whether the model is operating and improving in the desired way.

Efficiency and Velocity Benchmarks There are a few more, less defined measurements on how to assess a Graph's efficiency and speed.

The most prominent scale easily is the *Query Response Time*. It measures how fast a Knowledge Graph can find the response to a query. There are multiple sorts of tasks that can be given to the model; either single-entity queries, where the answer is one exact instance, multi-hop queries, where all instances connected to one certain entity are to be filtered out, or pattern matching queries, which measure how fast a Graph can identify patterns or find specific sub-

graphs. Depending on what the model is supposed to do, the lower limit of what is acceptable can vary, but in general, a model that is able to answer 1000 simple-entity queries in hundred milliseconds is considered to be efficient [42].

A similar benchmark is the *Throughput*. It assesses the number of queries the model can work on in one second, either from a single user or from multiple users. The higher this rate is, the better the Knowledge Graph can handle a fast, ongoing stream of tasks. Conventional indications of an acceptable Throughput are 100 queries in a second from a single user or 500 queries in a second from ten different users [26].

One last time related measurement scale is the *Recovery Time*, often combined with the *Fault Tolerance*. This number estimates how well a Knowledge Graph works and delivers meaningful outcomes, even if some of the links are faulty, and how long it takes to locate the incorrect triples, extract them from the Graph and mend the gaps left by it [44].

Another thing to take into consideration when evaluating a Knowledge Graph is the *memory usage and storage space* the model takes up. It is always important to know how much space a Knowledge Graph needs when it is currently inactive, and how much that increases once queries are run [37].

Scalability describes how well a Knowledge Graph reacts to the size of its data increasing. This is measurable by adding either sources to create new triples from or already complete triples into the Graph, and examining how the quality of the query responses changes. If outputs stay as semantically correct as before, and the new data can be accessed in a meaningful way, the model is robust to data increases [35].

One more metric in the organization of a Graph is the *Graph Traversal Efficiency*. It assesses how quick the system is in traversing through the graph, especially for more complex queries that afford multiple hops. If this takes too long with too many nodes along the route, developers might need to rethink the structure of the graph and look for alternative routes with lesser junctions to get to the outcome [40].

Graph Validity Benchmarks The last subject that helps to rate a Knowledge Graph is its *Explainability*. This metric was designed to assess how well a model can deliver explanations for the aggregated responses in order to provide integrity. In many applications, the reason why the Graph chose its outcome is just as important as the answer itself, and in those cases the system often gets asked which triples provided the base for its decisions. An area where explainability carries exceptional importance is once again Link prediction; when the model adds a report to why it predicts triples in a certain way, it is easier to understand how the model draws assumption and find the sources of incorrect conclusions. The downside to this metric is that no definite formulas or calculations exist, usually it can only be done through user studies; the measuring method is to ask users to rate the descriptions given, either with a percentage or by ranking the options [2].

Evaluation Method	Papers employing this method
Precision and Recall	[10], [17], [1], [19]
F1-score	[18], [19]
Specificity	[24], [1]
Accuracy to Gold Standard	[33], [6], [24], [31], [1], [23], [19]
Confidence Level	[22], [34]
Efficiency	[6], [36]
Velocity and Speed	[30]
Explainability	[25], [18], [19]

Table 6: Evalutaion Methods employed by papers

Table 6 provides an overview of the Benchmarks used in the studied papers.

To sum up, there are various ways to evaluate a Knowledge Graph. Different metrics are meant to assess different components of the Graph, meaning that combining two or more alternatives helps gaining a comprehensive view of the finished model.

6 Challenges and Biases in Crowdsourcing

Even though Hybrid Intelligence in Knowledge Graph creation has made impressive progress in the past decade, morphing from applications mainly used for knowledge aggregation to cases designed for everyday use, there are still a few challenges to be addressed, and, as far as possible, overcome. In the following section, these trials will be discussed, and the question how avoidable bias in Crowdsourcing really is will be examined.

6.1 Challenges in Project Implementation

The first hurdle for Knowledge Graphs constructed by Natural Language processing tools, is, of course, the fact that textual documents can often contain difficult wording which makes it hard for an NLP program to draw connections between entities or understand synonymous expressions. Language parsers are constantly improving, yet many of the projects that were examined struggled with having sentence structures too complex for the program to comprehend. An example is shown in the project designing a Knowledge Base for literary works [17]: some actions, like pretending to fall in love with somebody were featured in the books rather often, yet the model had difficulties grasping what that meant and failed to come up with a fitting name for those relations.

Another problem with Natural Language processing is the fact that texts containing more than one language can cause confusion for the program. This is evident, for instance, in the family of BERT-based Large Language Models; BERT is a language model that was introduced in 2018 by Google AI, and

was one of the first examples of an LLM. The original version solely learned from English texts, and therefore was unable to draw any information from documents written in other languages [43]. After the model turned out to be successful after its release, BERT became a baseline for language models, and a BERT-based version was composed for many other languages, such as French (CamemBERT) [8], Polish (Herbert) [27] or Russian (Rubert) [20]. An advantage of the BERT-family of models was that both the Latin and the Cyrillic alphabet don't leave any room for interpretation, other alphabets like Japanese, on the other hand, do not possess this unambiguity [7]; one character having multiple differing meanings creates a lot of additional difficulty for a Natural Language parser. All in all, it can be said that often a model is applicable for a certain language, as soon as another is involved, however, problems may arise.

A struggle that originates in the human side of the process is the individual interpretation of subtexts or explanations. This was once again shown in [17], the project where a Knowledge Graph for famous books was created. After selecting triples for the crowdworkers to verify, it became apparent that different people had different perceptions of the relations between characters. For instance, some people verified the relation-tag "dislikes" when it came to two people from a story, others, however, found the tag "hates" to be more fitting, showing that sometimes disagreements between the model and the workers can stem from people having different interpretations of the same source material.

A further problem that human participation entails, is the decision on how to find the optimal balance between experts and non-experts, since both bring along advantages as well as disadvantages.

Naturally, non-experts are easier to enlist in a project, especially when looked for at Crowdsourcing marketplaces like Amazon MTurk. However, critical knowledge might be missing in their work, both regarding the topic and the required technological basics. Especially in fields like medicine, technology or other sciences and niche branches, non-experts aren't able to deliver all the background knowledge they need for making a productive contribution. Furthermore, workers should possess a certain level of understanding of Knowledge Graphs; yet, the right classification of entities and categorization of datatypes is a complicated affair and very susceptible to errors, especially with non-specialists contributing to the process [1].

From this, it can be deducted that experts are certainly necessary for a balanced and trustworthy outcome of Crowdsourcing. Unfortunately, they are harder to recruit and customarily paid more, which of course complicates the process, specifically in case of a niche topic Graph.

As already mentioned, finding a balance between these two groups is a difficult but important deed. Multiple of the examined projects have shown that models work best with short, concise explanations of entities and links. However, struggles existed with both experts' and non-experts' descriptions; in [18], the project developed to detect incorrect posts about Covid-19 on social media for instance, expert definitions often were too complicated and not concise

enough for the model to learn something and draw meaningful conclusions. The same happened with explanations from non-experts, though those usually didn't hold enough information as opposed to too much.

Other projects, like [6], rely solely on the expertise of specialists, which means a lot of effort goes into sorting through the answers to locate spammers. For this, at least, some Crowdsourcing marketplaces offer a verification process for workers to prove they possess knowledge on the topic, but this, naturally, wastes a lot of otherwise usable budget.

As stated in [30], some topics are more fit for Crowdsourcing than others. The main challenge lies in singling out what part of the Knowledge Graph can be evaluated by non-experts, and where experts are definitively essential.

Different problems have shown up in the more recent projects. The model from [25] for instance mentions, that when the base Knowledge Graph is developed by humans, it's often difficult to find appropriate criteria for the Graph's outlines. The example they give for this phenomenon is the following: in this model, users query for information on a specific topic. The system supplies them with an answer, and the users examine whether the outcome is relevant for the question they asked, and whether the reply answers the question to their satisfaction; this feedback is sent back to the graph. An unforeseen challenge that came with this process, however, was the definition of the term relevancy. Some crowdworkers marked certain answers as relevant, other people, on the other hand, didn't. This once again showed, that certain words cannot be defined easily, even though it is crucial for a meaningful way of working. As mentioned above, human beings are always endangered of interpreting things in different ways; yet, having differing opinions of "just" a relation-tag, brings much less chaos into the feedback loop than the instructions already leaving room for interpretation.

Other struggles surfaced in the projects that aimed to identify fake news on social media, [18] and [19]. First, there is the problem of binary outcomes. The model examines a Tweet, for example, and then decides whether the information portrayed in it is factually correct or not. Since this is a binary yes or no decision, the depth of the choice gets lost. On most occasions, not every single thing about a post is untruthful, but usually just a single component, like a name, a date, or a number. This means that, even if only about a fourth of the tweet is wrong, the model will rate it as completely incorrect, and that of course diminishes the success of the system quite a bit. Developers have suggested to let the model split the tweets into sentences, or even phrases, to limit this problem and determine what percentage of the post is credible, but naturally, this would take the extra process in training the model to split texts into meaningful sections. In the end, they decided that for now, singling out social media contributions that contained errors was enough, regardless of how much truth might still be held in those posts, and that calculating and discovering the fractions of correctness in the tweets was going to be part of future work.

Second, the fact that algorithms for multi-modal Knowledge Graphs aren't able to pick up on human social clues posed a big problem as well, as already

shortly described in section 3.4 with the example of Donald Trump not being fond of Joe Biden. Humans are able to understand how subtle interactions convey how two people feel about each other, and they are able to include these sentiments in the Graph. Once it is the models turn to do this, however, things get a lot more complicated. Even though the system from [19] is able to match pictures of people to their respective entities, it does not yet have the ability to distinguish gestures or expressions from videos or photographs. Naturally, there already are models which can recognize feelings in pictures based on the positioning of facial features and how these positions are translated into pixel information. But even though these systems exist, developers claimed that this was the limit of the model they had created. After all, the main goal was to create a Knowledge Graph that was able to filter out incorrect information either in only-text tweets or ones that featured a photograph of the mentioned people. To tread into the area of including emotions and inter-human relationships in this finding process, was not part of the experiment anymore and put off for future projects.

6.2 Crowdsourcing Distortion through Biases

Aside from the challenges described above, there is always a certain bias to be found in Crowdsourcing. In [9], a project that was conducted in 2019, the role and inclusion of prejudice in crowdworkers' activities has been researched.

The first thing the authors noticed was, unsurprisingly, that a lot of bias is included due to the selection of the workers. They found out that most Crowdsourcing marketplaces have a majority of workers from the USA and India; of course there are also many participants from Europe or Asian first world countries, yet on average, more than half of the people working on the same microtask inhabit one of those two countries; with that, they are able to win every majority vote on a HIT. This, of course, is problematic, since India and the United States combined make up only about 20 per cent of the world's population [4], yet are represented with more than 50 per cent in the microtasks. As a consequence, distortions arise, especially if the topic of the Knowledge Graph demands a more balanced geographical compositions of workers.

Due to the fact that humans always bring cultural biases along, even if only subconsciously, HITs have to be created to be as neutral as possible. Since English is the most common and frequent language in the world wide web, the tasks are usually worded in that tongue, still, the experiments of the project have shown that different nations show different prejudices towards certain words, which of course is not desired by most developers. Other factors that empirically have an effect on the outcome are the influence of other workers and the arrangement of the answer possibilities. If workers are able to see the results of others before they enter their own solution, they often choose differently than they would have without any influence; due to this, authors advise developers to not let workers see others' input at all. Furthermore, the response layout can manipulate answers, too. Many times, when a yes or no question contains the option "I don't know", users selected named third option. However, if the task

was rephrased into a Multiple-choice question with various suggestions *and* the "I don't know"-option, workers went for one of the recommendations, which, naturally, is more helpful for the model to develop. One last element that has a considerable impact of the tasks' outcome is the search engine users are allowed to use during the process. Smaller, local engines sometimes list the results in a different order than worldwide ones do, which causes different durations of each of the HITs as well as different rankings of relevancy for the search results.

Lastly, the authors aimed to examine the influence of demographic differences in an experiment, carried out with one test group from the USA and one from India. For this, they revised three HITs to filter for bias in the answers. The first question was whether workers considered Catalonia to be an independent country, to which two thirds answered with yes and the remaining third with no. Tendentially, those who picked the positive answer were younger people and women, while older people and men gravitated towards the negative option. Furthermore, those who chose no tended to take more time for research and went lower in the search result page on their browser, signalling they didn't take the first source they could find to establish their answer. The authors do not supply their opinion on whether lower results are more trustworthy than those higher up.

The second task was to name the capital city of Israel. A bit more than 80 per cent of the workers agreed that the capital was Jerusalem, the rest chose Tel Aviv, and no significant demographic differences were noticable.

In the last task, a deepfake video of the pope was shown to users, who were asked to choose whether the actions in the video really happened and provide an answer for their reasoning. Only a bit more than half of the workers correctly identified the clip as fake, the rest believed it to be real. Here, the geographical location could explain the disagreement rather well; two thirds of the workers from India believed the video to be real, while two thirds of the Americans identified it as invalid. A second factor that seemed to have had an effect was age. Surprisingly though, many of the younger participants thought the clip was truthful and older people were more suspicious, instead of the other way around.

To sum up, it can be said that, while Hybrid Intelligence certainly has come a long way, there are still some battles that must be fought in the process, either because of limits or biases. As the authors of [9] stated, a majority vote is always a great way of trying to achieve the best result of a microtask possible, humans should still always remain critical when supervising a Crowdsourcing project.

7 Conclusion

Knowledge Graph Creation is a field of growing popularity nowadays, and can be assisted by various semi-automatic building approaches. The process of constructing a KG with the help of Crowdsourcing has changed quite a bit in the past decade, as have the areas the Graphs are applied in. In this thesis, we an-

swered the Questions of what the typical characteristics of KG creation methods are and how they can be classified, and how the collaboration of humans and machines have changed in the past years. For this, we analyzed various research papers about models built with the help of Crowdsourcing, provided an overview of multiple creation processes and tracked the changes cooperation between humans and machines has gone through.

We concluded that there are many differences to be found when building a Knowledge Graph:

- The first one being how the underlying Knowledge Base is created and how the source material and textual documents are parsed for data; one option is using a Natural Language parser to filter information from texts, another employs human workers who read through articles, posts or other origins to create triples.
- A second difference is how many of those links are selected for the Crowdsourcing process: either the entire Graph can be worked on or the model chooses the connections it wants to have verified, corrected or given alternatives for; in these cases the system works a lot with confidence levels to choose the best triples for microtasks.
- Adding to this, we analyzed how some models can be worked on directly in the application, and that those usually enable the user to annotate every single triple. Developers in other projects favored the use of Crowdsourcing marketplaces like CrowdFlower. Regarding of the virtual space, there are many ways to design microtasks, like multiple choice questions, looking for incorrect links and many more.
- We furthermore discussed the different levels of human activity and worked out three different approaches: in two of them, the Knowledge Graphs are created by using NLP and the model devising triples. Then crowdworkers either interact with the model and become part of a feedback loop or solve microtasks where the outcome is analyzed by the model. In the third option, the links are created by humans and the system does test runs which are overseen and assessed by humans.
- Lastly, we discovered that, compared to the early 2010s, projects have become a lot more fit for day-to-day applications, and that they no longer serve as just a place for Knowledge aggregation but as something that could play an active part in everyone's life.

After examining these differences, we also answered the Question of how to evaluate a Knowledge Graph and what potential challenges and biases could be. We provided an overview of the many methods to assess how accurate a model's triples can be, how fast it is able to work, how the trustworthiness of a model can be displayed and how efficiently the links are created. At last, we looked at challenges that might arise both on the human and the machine side of the

creation process, and displayed that it is also important to pick a well-balanced group of crowdworkers.

In conclusion, we gave an extensive overview over different creation processes that displays to Knowledge Graph developers how to design the perfect operation for their individual models.

References

[1] Maribel Acosta, Amrapali Zaveri, Elena Simperl, Dimitris Kontokostas, Fabian Flöck, and Jens Lehmann. Detecting linked data quality issues via crowdsourcing: A dbpedia study. *Semantic web*, 9(3):303–335, 2018.

[2] Mohammed Alshammari, Olfa Nasraoui, and Scott Sanders. Mining semantic knowledge graphs to add explainability to black box recommender systems. *IEEE Access*, 7:110563–110579, 2019.

[3] Anonymous. What is entity linking? *https://www.ontotext.com/knowledgehub/fundamentals/what-is-entity-linking/*, Knowledge Hub, 2024.

[4] Anonymous. World population clock. *https://www.worldometers.info/world-population/: :text=8.2*

[5] Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. A survey on evaluation of large language models. ACM Transactions on Intelligent Systems and Technology*, 15(3):1–45, 2024.*

[6] Xu Chu, John Morcos, Ihab F Ilyas, Mourad Ouzzani, Paolo Papotti, Nan Tang, and Yin Ye. Katara: A data cleaning system powered by knowledge bases and crowdsourcing. In Proceedings of the 2015 ACM SIGMOD international conference on management of data*, pages 1247–1261, 2015.*

[7] Soumendu Das and Sreeparna Banerjee. An algorithm for japanese character recognition. International Journal of Image, Graphics and Signal Processing*, 7(1):9, 2014.*

[8] Cyrile Delestre and Abibatou Amar. Distilcamembert: a distillation of the french model camembert. arXiv preprint arXiv:2205.11111*, 2022.*

[9] Gianluca Demartini. Implicit bias in crowdsourced knowledge graphs. In Companion Proceedings of The 2019 World Wide Web Conference*, pages 624–630, 2019.*

[10] Gianluca Demartini, Djellel Eddine Difallah, and Philippe Cudré-Mauroux. Large-scale linked data integration using probabilistic reasoning and crowdsourcing. The VLDB Journal*, 22(5):665–687, 2013.*

[11] *Djellel Eddine Difallah, Michele Catasta, Gianluca Demartini, Panagiotis G Ipeirotis, and Philippe Cudré-Mauroux. The dynamics of micro-task crowdsourcing: The case of amazon mturk. In* Proceedings of the 24th international conference on world wide web*, pages 238–247, 2015.*

[12] *Junyang Gao, Xian Li, Yifan Ethan Xu, Bunyamin Sisman, Xin Luna Dong, and Jun Yang. Efficient knowledge graph accuracy evaluation.* arXiv preprint arXiv:1907.09657*, 2019.*

[13] *Ralph Grishman. Information extraction.* IEEE Intelligent Systems*, 30(5):8–15, 2015.*

[14] *Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard De Melo, Claudio Gutierrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, et al. Knowledge graphs.* ACM Computing Surveys (Csur)*, 54(4):1–37, 2021.*

[15] *Jeff Howe et al. The rise of crowdsourcing.* Wired magazine*, 14(6):176–183, 2006.*

[16] *Ece Kamar. Directions in hybrid intelligence: Complementing ai systems with human intelligence. In* IJCAI*, pages 4070–4073, 2016.*

[17] *Sarath Kumar Kondreddi, Peter Triantafillou, and Gerhard Weikum. Combining information extraction and human computing for crowdsourced knowledge acquisition. In* 2014 IEEE 30th International Conference on Data Engineering*, pages 988–999. IEEE, 2014.*

[18] *Ziyi Kou, Lanyu Shang, Yang Zhang, and Dong Wang. Hc-covid: A hierarchical crowdsource knowledge graph approach to explainable covid-19 misinformation detection.* Proceedings of the ACM on Human-Computer Interaction*, 6(GROUP):1–25, 2022.*

[19] *Ziyi Kou, Yang Zhang, Daniel Zhang, and Dong Wang. Crowdgraph: A crowdsourcing multi-modal knowledge graph approach to explainable fauxtography detection.* Proceedings of the ACM on Human-Computer Interaction*, 6(CSCW2):1–28, 2022.*

[20] *Yuri Kuratov and Mikhail Arkhipov. Adaptation of deep bidirectional multilingual transformers for russian language.* arXiv preprint arXiv:1905.07213*, 2019.*

[21] *Thomas D LaToza, W Ben Towne, Christian M Adriano, and André Van Der Hoek. Microtask programming: Building software with a crowd. In* Proceedings of the 27th annual ACM symposium on User interface software and technology*, pages 43–54, 2014.*

[22] *Chunhua Li, Pengpeng Zhao, Victor S Sheng, Xuefeng Xian, Jian Wu, and Zhiming Cui. Refining automatically extracted knowledge bases using crowdsourcing.* Computational Intelligence and Neuroscience*, 2017(1):4092135, 2017.*

[23] Juan Antonio Lossio-Ventura, William Hogan, François Modave, Yi Guo, Zhe He, Xi Yang, Hansi Zhang, and Jiang Bian. Oc-2-kb: integrating crowdsourcing into an obesity and cancer knowledge base curation system. BMC medical informatics and decision making*, 18:115–127, 2018.*

[24] Yun Lou, Samson W Tu, Csongor Nyulas, Tania Tudorache, Robert JG Chalmers, and Mark A Musen. Use of ontology structure and bayesian models to aid the crowdsourcing of icd-11 sanctioning rules. Journal of Biomedical Informatics*, 68:20–34, 2017.*

[25] Mengtao Lyu, Xinyu Li, and Chun-Hsien Chen. Achieving knowledge-as-a-service in iiot-driven smart manufacturing: A crowdsourcing-based continuous enrichment method for industrial knowledge graph. Advanced Engineering Informatics*, 51:101494, 2022.*

[26] Jason Mohoney, Anil Pacaci, Shihabur Rahman Chowdhury, Ali Mousavi, Ihab F Ilyas, Umar Farooq Minhas, Jeffrey Pound, and Theodoros Rekatsinas. High-throughput vector similarity search in knowledge graphs. Proceedings of the ACM on Management of Data*, 1(2):1–25, 2023.*

[27] Robert Mroczkowski, Piotr Rybak, Alina Wróblewska, and Ireneusz Gawlik. Herbert: Efficiently pretrained transformer-based language model for polish. arXiv preprint arXiv:2105.01735*, 2021.*

[28] Sujatha Mudadla. Difference between knowledge graph and knowledge base. medium.com*, 2023.*

[29] Prakash M Nadkarni, Lucila Ohno-Machado, and Wendy W Chapman. Natural language processing: an introduction. Journal of the American Medical Informatics Association*, 18(5):544–551, 2011.*

[30] Allard Oelen, Markus Stocker, and Sören Auer. Creating and validating a scholarly knowledge graph using natural language processing and microtask crowdsourcing. International Journal on Digital Libraries*, 25(2):273–285, 2024.*

[31] Prakhar Ojha and Partha Talukdar. Kgeval: Accuracy estimation of automatically constructed knowledge graphs. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1741–1750, 2017.*

[32] Heiko Paulheim. Knowledge graph refinement: A survey of approaches and evaluation methods. Semantic web*, 8(3):489–508, 2017.*

[33] Perrine Pittet and Jérôme Barthélémy. Exploiting users' feedbacks-towards a task-based evaluation of application ontologies throughout their lifecycle. In International conference on knowledge engineering and ontology development*, volume 2, pages 263–268. SCITEPRESS, 2015.*

[34] André Pomp, Johannes Lipp, and Tobias Meisen. You are missing a concept! enhancing ontology-based data access with evolving ontologies. In 2019 IEEE 13th International Conference on Semantic Computing (ICSC)*, pages 98–105. IEEE, 2019.*

[35] Sumit Purohit, Nhuy Van, and George Chin. Semantic property graph for scalable knowledge graph analytics. In 2021 IEEE International Conference on Big Data (Big Data)*, pages 2672–2677. IEEE, 2021.*

[36] Yifan Qi, Weiguo Zheng, Liang Hong, and Lei Zou. Evaluating knowledge graph accuracy powered by optimized human-machine collaboration. In Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1368–1378, 2022.*

[37] Zhixin Qi, Hongzhi Wang, and Haoran Zhang. A dual-store structure for knowledge graphs. IEEE Transactions on Knowledge and Data Engineering*, 2021.*

[38] Junyuan Quan and Ning Wang. An optimized task assignment framework based on crowdsourcing knowledge graph and prediction. Knowledge-Based Systems*, 260:110096, 2023.*

[39] Yves Raimond, Tristan Ferne, Michael Smethurst, and Gareth Adams. The bbc world service archive prototype. Journal of web semantics*, 27:2–9, 2014.*

[40] Hongyu Ren, Hanjun Dai, Bo Dai, Xinyun Chen, Denny Zhou, Jure Leskovec, and Dale Schuurmans. Smore: Knowledge graph completion and multi-hop reasoning in massive knowledge graphs. In Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1472–1482, 2022.*

[41] Marta Sabou, Stefani Tsaneva, Miriam Fernandez, María Poveda-Villalon, and Mari Carmen Suárez-Figueroa. Human-centric Evaluation of Semantic Resources:A Systematic Mapping Study Protocol and Data, March 2024.

[42] M Srikanth and RNV Jagan Mohan. Machine learning for query processing system and query response time using hadoop. IJMTST, Aug*, 2020.*

[43] I Tenney. Bert rediscovers the classical nlp pipeline. arXiv preprint arXiv:1905.05950*, 2019.*

[44] Rui Xie, Long Chen, Wei Ye, Zhiyu Li, Tianxiang Hu, Dongdong Du, and Shikun Zhang. Deeplink: A code knowledge graph based deep learning approach for issue-commit link recovery. In 2019 IEEE 26th International Conference on Software Analysis, Evolution and Reengineering (SANER)*, pages 434–444. IEEE, 2019.*

8 Appendix A

Excel Data Extraction: https://docs.google.com/spreadsheets/d/1SxmtB5Wc9att3zDNEaVALAm7Y5W55CxW
edit?usp=drive_link&ouid=109471385235985324293&rtpof=true&sd=true