Human-guided exploration of datasets

Mehrdad Farokhnejad

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ GRENOBLE ALPES

Spécialité : Informatique

Arrêté ministériel : 25 mai 2016

Présentée par

Mehrdad FAROKHNEJAD

Thèse dirigée par **Genoveva VARGAS-SOLAR**,CR, Université Grenoble Alpes
et co-encadrée par **Javier Alfonso ESPINOSA OVIEDO**, UGA

préparée au sein du **Laboratoire Laboratoire d'Informatique de Grenoble**
dans **l'École Doctorale Mathématiques, Sciences et technologies de l'information, Informatique**

Exploration interactive de collections de données guidée par l'humain

Human-guided exploration of datasets

Thèse soutenue publiquement le **15 octobre 2021**,
devant le jury composé de :

Madame CLAUDIA RONCANCIO
PROFESSEUR DES UNIVERSITES, GRENOBLE INP, Présidente
Monsieur LAURENT D'ORAZIO
PROFESSEUR DES UNIVERSITES, UNIVERSITE RENNES 1, Rapporteur
Madame KARINE ZEITOUNI
PROFESSEUR DES UNIVERSITES, UNIVERSITE PARIS-SACLAY, Rapporteure
Monsieur BERND AMANN
PROFESSEUR DES UNIVERSITES, SORBONNE UNIVERSITE, Examinateur
Madame TANIA CERQUITELLI
PROFESSEUR ASSISTANT, Politecnico di Torino, Examinatrice
Monsieur VASSILIS CHRISTOFIDES
PROFESSEUR DES UNIVERSITES, ENS ELECTRONIQ APPLICAT CERGY, Examinateur
Madame MARIA-ESTHER VIDAL
PROFESSEUR, Gottfried Wilhelm Leibniz Univ. Hannover, Examinatrice

Acknowledgements

I thank Prof. Laurent D'Orazio, University of Rennes, and Prof. Karine Zeitouni, UVSQ - University Paris Saclay, for accepting reviewing this work. I thank Prof. Bernd Amann, University Sorbonne, Prof. Tania Cerquitelli, Politecnico di Torino, Prof. Vassilis Christophides, Ecole National Supérieure d'Electronique et de ses Applications (ENSEA), Prof. Claudia Roncancio Grenoble Institute of Technology - ENSIMAG and Prof. Maria Esther Vidal, Informationszentrum Technik und Naturwissenshaften Universitaets Bibliothek, for accepting participating as members of the jury.

I want to express my sincere gratitude to my supervisor, Dr. Genoveva Vargas-Solar, Principal Scientist at CNRS, for her excellent mentoring, guidance and support through my PhD journey. She helped me through many difficulties I had during these years. Without her advice and guidance, I do not think this dissertation would have been possible. I learned a lot from her knowledge and professionalism. During the PhD process, she always encouraged me to express my ideas. It is unimaginable how much time and effort she had to spend to discuss, proofread and correct all my works. I would also like to thank my thesis co-advisor, Dr. Javier A. Espinosa-Oviedo, postdoctoral researcher at University Lumière Lyon 2, for all his helpful feedback on my work. His thoughtful comments and detailed feedback helped me a lot.

I want to thank the Ministry of Science, Research and Technology of Iran for providing the funding to undertake my research. I would also like to extend my thanks to the faculty members, office staff, and technical staff of the Laboratory of Informatics of Grenoble (LIG) for their support during all these years.

I would also like to extend my thanks to express my deepest love and gratitude to my family members. Special appreciation goes to my wife, Mobina, whose love, encouragement, understanding and patient support sustained me in completing my graduate degree.

Last but not least, I would like to praise and thank God, the almighty, who has granted countless blessings, knowledge, and opportunity to the writer so that I have been finally able to accomplish the thesis.

Abstract

Data exploration aims to guide the understanding of data collections and define the type of questions that can be asked on top, often in interactive exploration processes. Data exploration deals with raw digital data collections coping with the uncertainty of data content and analysis where query results cannot be necessarily correct and complete (i.e., results consisting in all the data tuples respecting requirements expressed by a question). Data exploration engines will be next-generation systems promoting a new querying philosophy that gradually converges into queries that can exploit raw data collections that cope with data explorers (i.e., users) expectations.

This thesis proposes HILDEX, a human-in-the-loop based data exploration system that enables users to explore textual data collections by gradually refining queries and associated results. Textual data collections are pre-processed using Machine Learning and Artificial Intelligence text processing algorithms.

HILDEX implements exploration algorithms proposed in this work (query morphing, query-by-example, queries-as-answers) that allow refining an initial query by considering the content of the collections to be explored to increase the possibility to explore the data better. Therefore, HILDEX proposes a workflow to explore texts by analysing data samples obtained by queries that can be refined through human in the loop-based tasks. Partial exploration results are assessed through metrics (precision, similarity) and information that explains why some documents are contained in these results. By exploring documents in partial results, explanations and metrics, the user can decide to continue interacting with HILDEX for rewriting queries until she is satisfied with both queries and results. The algorithms and HILDEX have been experimented on data related to crises in urban computing and the exploration of information on COVID-19.

Résumé

L'exploration des données vise à guider la compréhension des collections de données et à définir le type de questions qui peuvent être posées dessus, souvent dans le cadre de processus d'exploration interactifs. questions qui peuvent être posées dessus, souvent dans le cadre de processus d'exploration interactifs. L'exploration de données traite l'exploration de données porte sur des collections de données numériques brutes et fait face à l'incertitude du contenu et de l'analyse des données. résultats des requêtes ne peuvent pas être nécessairement corrects et complets (c'est-à-dire des résultats comprenant tous les tuples de données respectant les exigences exprimées par une question). exigences exprimées par une question). Les moteurs d'exploration de données seront des systèmes de nouvelle génération promouvant une nouvelle philosophie d'interrogation qui converge progressivement vers des requêtes capables d'exploiter des données brutes. des collections de données qui répondent aux attentes des explorateurs de données (i.e., les utilisateurs).

Cette thèse propose HILDEX, un système d'exploration de données basé sur le "human-in-the-loop" qui permet aux utilisateurs d'explorer des collections de données textuelles en raffinant progressivement les requêtes et les résultats associés. Les collections de données textuelles sont prétraitées à l'aide d'algorithmes de traitement de texte de type Machine Learning et Intelligence Artificielle.

HILDEX propose des algorithmes d'exploration (query morphing, query-by-example, queries-as-answers, query generation) qui permettent de raffiner une requête initiale en considérant le contenu des collections à explorer pour augmenter la possibilité de mieux explorer les données. Par conséquent, HILDEX propose un flux de travail pour explorer des textes en analysant des échantillons de données obtenus par des requêtes qui peuvent être affinées par des tâches basées sur le "human-in-the-loop". Les résultats de l'exploration partielle sont évalués à l'aide de métriques (rappel, précision, score F1) et d'informations expliquant pourquoi certains documents sont contenus dans ces résultats. En explorant les documents dans les résultats partiels, les

explications et les métriques, l'utilisatrice peut décider de continuer à interagir avec HILDEX pour réécrire des requêtes jusqu'à ce qu'elle soit satisfaite à la fois des requêtes et des résultats. Les algorithmes et HILDEX ont été expérimentés sur des données relatives aux crises naturelle (informatique urbaine) et à l'exploration des informations sur la COVID-19.

Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

The evolution in querying, information retrieval, and human-computer interaction has led to the shift of interest from the traditional *query-response* paradigm to actual human guided-data exploration systems. *Data exploration* promotes a new querying philosophy that gradually converges into queries that can be used to exploit raw data collections according to data explorers (i.e., users) expectations [1]. A traditional query-response system is limited to fact-finding, whereas the current paradigm requires gathering, understanding, and using insightful information by the users [1]. Users' intention helps to navigate through the unknown data, formulate queries and find the desired information. In most of the occurrences, user feedback acts as relevance criteria for the following query search iteration [2, 3]. Such novel requirements of modern exploration driven processes call to rethink data querying processes.

Interactive data exploration techniques assist users in extracting interesting insights from data collections and several techniques have been proposed with different perspectives [1]. For example, query steering [4] relies on the user's feedback to provide suggestions. QueRIE [5] recommends the most likely queries, and the YMALDB framework [6] allows users to discover highly correlated and similar tuples to the original query's results, although these discovered tuples are not among the original query's results. Additionally, *query-by-example* techniques [7, 8] receive as input various examples of expected results to prevent the user formulating a concrete and precise query. Finally, the work in [9] helps explorers interactively and efficiently navigate through large data spaces by clustering data.

In general, existing approaches address the expression of queries for retrieving information in a query-answer scenario where results must match the query. Exploration, instead, is intended to extend the resulting scope that matches a query for identifying items (i.e., documents) that are possibly interesting. Other approaches consider exploration as a quantitative and classification task, where the result provides statistics, frequent terms, and classes of items clustered according to salient features. However, these approaches do not provide detailed insight into the content of data collections. We believe that existing approaches must be calibrated to promote the rewriting of queries and associate them to samples of the data collections they target to let the user understand the content and decide whether the queries fulfil her requirements.

Data exploration is a complex process, as the user has to decide which exploratory operation to employ at each step of the exploratory process. None of the existing systems has considered creating a hybrid workflow that combines different data exploration techniques. Existing exploration systems have limited provisions to help users reformulate their queries as they evolve with the search progression.

1.1 Motivating example

To illustrate our vision of data exploration, let us consider the following example motivated by an exploration requirement that emerges in disaster management scenarios (e.g., earthquake, flooding, fire), where social media data are produced describing people's situation, observations, requests, and offer of voluntary assistance during the life-cycle of a disaster. These data must be explored so that different actors can organize relief, resilience and duty of memory actions [10].

During such life-threatening emergencies, affected and vulnerable people, humanitarian organizations, and other concerned authorities search for information to help prevent a crisis. Nobody has control over the type of data exchanged by actors, yet it is crucial to make critical decisions like saving lives, searching people, providing shelter, and medical assistance. For this reason, it is required to explore past and present in an agile manner to find hints to make decisions and act individually and collectively. Considering that data collections have been produced within social networks, multi-media data can include reports on architectural and building environment damages and also volunteers informing that they have answered calls for help (see Figure 1.1).

The question is whether these data collections can help to:

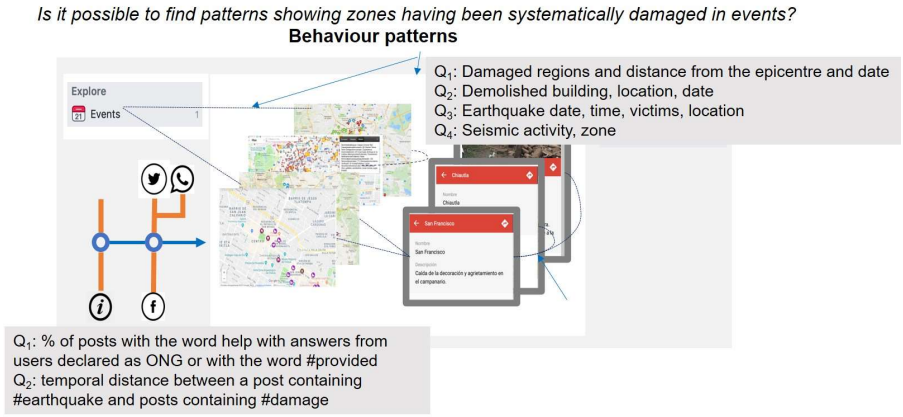

Figure 1.1: Data exploration in disaster management.

- Correlate events and actions. For instance, given a post asking for help, is it possible to know whether actions have already been taken? Since when and whether the problem has been solved?
- Information patterns. For instance, is it possible to find patterns showing which zones have been systematically damaged in other events? Is there more risk and help required in those regions?
- New questions. For instance, what is the type of help still being required after a day of the event?

Note that these questions are not asking for results but asking for assistance on how to ask them on top of data collections so that people can potentially best explore the data. The general question is *how can I express my query to expect to receive the best guidance to act? Is my query pertinent to be asked given the data I can have access to?* Data exploration techniques can help to assist in the expression of queries that can potentially explore disaster data collections.

While research has been conducted on how to provide alternative queries in general web searches [11,12], little work has been done about the suggestion for domain-specific exploration. Query exploration techniques should be designed for the unique search characteristics of this domain. For example, disaster data exploration is typically recall-oriented [10]. Other types of queries are rather precision-oriented.

The main difference of this task from ad-hoc retrieval (e.g., web search) is that exploration tends to be recall-oriented, with users examining many retrieval results. Accordingly, exploration processes are often long and

repeatable. A user issues several queries until the obtained results are satisfactory.

1.2 Problem statement: Exploring massive textual raw data collections

The fundamental assumptions of traditional data management systems are that users have knowledge about the structure and content of the dataset and have a clear understanding of what questions (queries) are to be asked. However, these assumptions are becoming less accurate as the volume of data increases. In addition, users are often not familiar with the underlying database and are not sure which patterns they want to find and can be exploitable to answer their questions. They will only realize something is interesting once they have discovered it. In this setting, there are no clear indications about how the users should formulate their queries. Instead, they may want to navigate through a subspace of the dataset to find items of interest or may want to see a few samples, provide yes/no feedback, and expect the system to find more similar objects [13].

Interactive data exploration promotes a new querying philosophy that gradually converges into queries that can exploit raw data collections according to data explorers (i.e., users) expectations. Interactive data exploration aims to guide the understanding of data collections with different rawness degrees and define the questions that can be asked on top, often through interactive exploration processes. This work introduces human-guided data exploration defining exploration operations that result in different types of queries.

Our research addresses the problem of (i) assisting data consumers to effectively and interactivity explore textual data collections for computing query sets that can let them better exploit raw data collections and (ii) discovering the type of questions that such data collections can answer within an exploration purpose.

For example, given a user intention expressed using terms and constraints on raw data collections, the proposed approach defines a human-in-the-loop data exploration pipeline that:

- computes different types of queries that can be asked on top of data that cope to user's expectations;
- integrates an interaction strategy with the user for refining intentions based on the proposed queries;

– starts the loop again until the queries proposal converges with user expectations.

1.2.1 Objective and approach

The general objective of our work is to propose a human-guided data exploration approach for assisting textual data exploration using different complementary techniques.

1. Model human-guided data exploration pipelines as workflows that can be then enacted in an ad-hoc environment.
2. Design and implement a tool to explore data collections to assist data consumers to identify which types of conclusions can potentially be derived.
3. Design an experiment for exploring risk and disaster management.

The principle of our approach is the following: given an initial Boolean keyword query, (1) rewrite the query to provide sets of queries to exploit data collections better (2) define an interactive and iterative process where the user selects the queries that produce results that are close to her requirements.

Our approach relies on the hypothesis that textual data collections are processed for producing vocabularies, identifying frequent terms and contextual representation to ease the exploration tasks. The approach uses this information to rewrite queries and explore textual data collections.

Given a Boolean keyword query, our approach rewrite it and provide more queries that can potentially give better results. Query rewriting is done by applying complementary techniques that are combined into a human-in-the-loop based exploration workflow. A data exploration framework implements this workflow.

1.2.2 Contribution

The first contribution of this work is the definition and implementation of three exploration algorithms: *query morphing*, *queries-as-answers* and *query-by-example*. These algorithms rely on existing information retrieval techniques, but they change the underlying principle. Instead of rewriting queries for retrieving documents that match a query as precisely as possible, the proposed algorithms extend the matching scope of an initial query through alternative queries. The objective is to let the user explore a data collection content to determine to which extent it can fulfil her requirements.

We have proposed HILDEX, a human-guided data exploration framework. HILDEX combines these exploration algorithms into a workflow that results in different types of queries used to explore textual data collections content. The idea is to include the interaction with the user to propose and adjust the type of questions that can be answered.

In general, the HILDEX data exploration workflow:

– defines a loop where given a user intention expressed using terms and raw data collections, the exploration strategies propose different types of possible queries that can be asked on top of data and that potentially correspond to user's expectations;

– interacts with the user for refining intentions based on the proposed queries and starts the loop again until the queries proposal converges with user expectations.

1.3 Document organisation

The dissertation is organised as follows:

- *Chapter 2* proposes a state of the art on data exploration. It describes the techniques and systems that have been proposed for addressing different data exploration requirements. It proposes a query exploration taxonomy that summarises the type of exploration queries that have been addressed in existing approaches, including those that include human-in-the-loop. The taxonomy also classifies systems and types of data collections associated with exploration tasks.

- *Chapter 4* introduces three exploration algorithms proposed in this work that were calibrated for textual data collections: query morphing, queries-as-answers and query-by-example. The chapter introduces preliminary concepts underlying the algorithms. Then it describes the general principle of each algorithm and shows examples that illustrate their use.

- *Chapter 3* introduces HILDEX the human-in-the-loop data exploration framework that we propose. The chapter first describes an interactive data exploration workflow which is the kernel of the framework. The workflow combines different query rewriting techniques that enable data explorers to find the appropriate queries to explore data collections taking full advantage of their content.

- *Chapter 5* introduces an experimental setting that we used for validating and assessing the HILDEX exploration workflow. The validation is

based on three use cases, showing different exploration requirements performed on crisis management and COVID-19 textual data collections.

- *Chapter 6* summarises the work, its contributions, and discusses the scope and open issues. Finally, the chapter enumerates and describes the technical and scientific perspectives of the work.

CHAPTER 2

Data Exploration: Queries, Techniques and Systems

Exploration is one of the ways to accrue knowledge about the world and its nature. It describes the act of becoming familiar with something by testing or experimenting.

As unprecedented volumes of heterogeneous data are automatically collected and accessible, data collections have become less and less familiar to consumers. Exploratory search emerges as the process of gradual discovery and understanding of the portion of the data that is pertinent to an oftentimes vague user's information need [14]. This chapter introduces an analytic study of the academic and industrial work and systems that have addressed data exploration. The objective of the chapter is to build a cartography of data exploration definition, problems and solutions. This cartography serves to identify open issues and determine the perimeter of our work compared to existing solutions.

Accordingly, the chapter is organised as follows. Section 2.1 analyses and compares data exploration algorithms and methods. Section 2.2 summarises existing data exploration systems and tools concerning the functions and principles they propose for exploring data. Section 2.3 proposes a classification of query exploration methods, systems and tools based on a systematic mapping that we performed (see Appendix A.1 for a complete version of the study). The classification serves as a cartography of the different perspectives of data exploration proposed in different disciplines like databases, data science and information retrieval. Finally, Section 2.4 concludes the chapter

and discusses open issues identified that our work addressed.

2.1 Data exploration methods

The term data exploration refers to a data user finding her way through large amounts of data to gather necessary information. From the statistics point of view [15], through exploratory data analysis, a researcher explores the data in many possible ways (e.g., graphical tools like boxplots or histograms) to gain knowledge from the way data are displayed.

Authors in [16] define data exploration as a step-by-step "conversation" of a user and a system that "help each other" to refine the data exploration process, gathering new knowledge that concretely fulfils the user needs. The process is seen as a conversation since the system answers the user's requests and suggests possible actions to help the user focus on the exploratory session. Thus, data exploration entails using a wide range of techniques like statistics and data analysis, query suggestion, advanced visualization tools, etc.

There are two possible perspectives to classify data exploration methods: (i) depending on whether the processes is guided by queries, (ii) or by processing data collections content with models. In both cases they can include human in the loop techniques. The objective to put the user in the centre of the process is understand her requirements and profile and calibrate the exploration process accordingly.

- In the first case, the expected result of the process is a query or set of queries that can be evaluated upon the collection with good expected assessment scores (precision, recall).
- In the second case the objective is to acquire a good understanding of the data collection content by processing it with different data mining, statistical and machine learning methods that yield to a thorough representation in quantitative, qualitative, structural and even semantic terms. For instance, the distribution of the values, the percentage of missing and null values, the structure of the data (e.g., column names and types for tabular data), the vocabulary and concepts present within the content.

2.1.1 Queries-based exploration

A critical challenge in the data exploration process is discovering and issuing the "right" query. Identifying queries that users are most interested in is critical for supporting exploratory data analysis. Even if database systems

can run complex queries over large datasets, discovering helpful information remains a big challenge. Users unfamiliar with the database might overlook queries that retrieve interesting data or may not know what parts of the database provide useful information. This issue hinders data exploration. Table 2.1 summarises and illustrates queries-based data exploration methods.

Table 2.1: Query-based data exploration methods.

1. Query Based Data Exploration	
1.1 Query Expansion	New related terms [17], keyword [18], supervised query expansion (adaptive & cost constrained) [19], faceted search [20, 21], measure driven [22], probabilistic [23], result set post-processing [24], query morphing [25]
1.2 Queries as Answers	Log based [26], rewriting with auto-completion [27]
1.3 Multi-scale Queries	

There is a massive amount of data available, and it is growing exponentially. Often, a web search does not yield relevant results. Query-based exploration techniques play an essential part in fetching relevant results. Users usually submit queries containing ambiguous terms that can retrieve documents that do not correspond to the field of research of the user. Besides, users submit short queries and that the average length of web queries is two words or less [28]. Short queries usually do not have enough words to cover useful search terms and thus affect the search performance negatively. In many approaches, queries are keyword expressions; however, keywords are not always a good descriptor of the needed information. Therefore it was essential to propose solutions to make it easy to retrieve the required information by rewriting queries using many different strategies.

Query expansion (QE) approaches (1.1 in table 2.1) rewrite the query with terms extracted from various information sources. The goal of Global Expansion techniques [17] is to complete the user query using the global information extracted from the whole explored corpus (e.g., WordNet). Local Expansion techniques, extract the expansion terms from the top-ranked documents [18].

Popular techniques expand the initial query by adding new related terms. This strategy can also involve selective retention of terms from the original query. The expanded/reformulated query is then used to retrieve more relevant results. The survey in [29] defines a query expansion process consisting of the following steps: data processing (performed on data collections and queries), term extraction, terms weights and ranking, term selection and query reformulation. Weighting and ranking terms can be:

- one to one: correlates each expanded term to at least one query term using linguistic associations, for instance, a stemming algorithm, thesauri (e.g. Wordnet);
- one to many: correlates each expanded term to many query terms;
- feature distribution of top-ranked documents: deals with the top retrieved documents from the initial query and consider the top-weighted terms from these documents;
- query language modeling: constructs a statistical model for the query and chooses expansion terms having the highest probability.

Most major search engines provide query expansion functionality in the form of query auto-completion. The suggested expansions are ranked in a query-independent, and data-agnostic manner, such as based on their popularity in query logs [27]. The assumption is that similar words that frequently appear in documents belong to the same subject. Thus, similar documents formed a cluster. In existing methods, clusters can group terms (use thesauri of terms) and documents (thesauri of documents).

Query expansion has been enhanced by applying supervised learning and adaptability properties. Supervised Query Expansion (SQE) [19] is a corpus-specific query expansion technique that consists of stages to predict whether a query is suitable for SQE or not (Adaptive Expansion Decision (AED) and choose a subset of features for supervised learning (Cost Constrained Feature Selection (CCFS)).

Faceted search [20] enables the progressive refinement of a keyword-query result. The principle is that each document is associated with a set of well-defined categorical attributes (meta-data) referred to as facets. The meta-data domains are usually organized in a hierarchy. Then, the result of a vanilla keyword query is refined by "drilling down" the facet hierarchies. This interactive process places and gradually tightens constraints on the meta-data, allowing to identify and focus on a fraction of the documents that satisfy a keyword query. Faceted search has similar aims w.r.t. database exploration. The work introduced in [21] improves faceted search using "dynamic" facets extracted from the content of the documents: a result set can be further refined using frequent phrases appearing in the documents comprising the result.

Measure-driven Keyword-Query Expansion [22] is a searching model similar to faceted search that enables the progressive refinement of a keyword-query result. The refinement process is driven by suggesting interesting expansions of the original query with additional search terms. This query-

driven and domain-neutral approach employ "surprising" word co-occurrence patterns and (optionally) numerical user ratings to identify significant top-k query expansions and allow to focus on a particularly interesting subset of the original result set.

The work in [23] uses Query Expansion Methods and information of interactions between the users and the system. The work proposes a Probabilistic Query Expansion method that searches within a set of candidate terms for the most relevant terms for the initial query to expand. The output is a table of terms that are candidates for expanding the user query and their values of correlation with the whole query.

The general objective of result set postprocessing is to combine result cases. Once a query has been completed, resulting in one or more sets of items, the set can be post-processed to summarise the results or otherwise modify them. For that, the result set is not shown to the user as is, but passed through a straightforward statistics post-processor to derive informative results. Result set post-processing assumes an array of simple statistical information such as min, max, and mean to be more beneficial, especially on massive datasets [24]. For larger results, the data distribution may be further elicited by selective sampling. Histograms could also be constructed such that, for example, the number of bins shown fits the screen, while the bounds are derived based on the value deviation within each bin.

Query morphing is an approach for reformulating queries. The principle is that a user iteratively transforms her search request to gain relevant information and stops when sufficient information is met. User's exploration tasks can be categorized into three types: [25]:

- Lookup: tasks can relate traditional search with carefully formulated queries that yield precise results. For exploratory search tasks, lookup seeks more involvement of the user beyond just a query specification.

- Learn: a user tries to acquire knowledge about the domain and better understand it. It is an iterative process that simulates analogical thinking and relates users' experiences to return results. Reformulating queries and comparing results take much time in learning tasks.

- Investigate: reduces the knowledge gap and transforms existing data into new knowledge.

Result set post-processing and query morphing go on the premise that the user probably does not need the exact answer to her query.

Query-as-answers (1.2 in table 2.1) tackles the lack of knowledge a user may have on the dataset. Rather than responding to a priori bad queries (too long to run, too many tuples), the technique proposes a new list of queries based on the database's information optimized in various directions. The challenge of this solution is identifying bad queries, which can be done using the optimizer statistical information or massive scientific databases and identifying interesting queries to return. This task could be done using a list of frequently used queries and returning them based on user feedback.

Multi-scale queries As the user becomes more knowledgeable and confident in the data she uses and the direction of her queries, she will be willing to commit more resources and data to her search (i.e., start small and go big). Therefore, databases are a priori partitioned according to collection and databases into many files. Data partition means that queries can be performed on subsets of the files. The multi-scale queries method proposes, rather than one performing a query on the whole databases, to split it into multiple queries executed on different fragments of the database and then to perform a union of those queries. Multi-scale querying (1.3 in table 2.1) allows for a natural scaling of the size of the query as the user gets more confident in her query.

2.1.2 Model-based data exploration

Table 2.2 summarises and illustrates model-based data exploration methods. These methods can use (descriptive) statistics to produce quantitative insight into data. They can also extract knowledge utilizing data mining techniques [30]. Finally, they can apply visualization methods for exploring data.

Table 2.2: Model based exploration methods.

2. Model Based Data Exploration	
2.1 Statistics	Histogram (algebra) [31], sampling speculative execution [1], cube exploration [32], INDIANA [33], [15]
2.2 Knowledge Extraction	ADESCA [34], Helix [35], READ [36], EPIQUE [37], HARE [38]
2.3 Visualisation	DataWrapper[1], Datapine[2], Zenvisage [39], Answerminer[3], VIQS [40], GraphTQL [41], [16], DIVE [42], [43], [44]

Statistics based exploration. To summarize the content of large relations and permit efficient retrieval of approximate query results, authors in [31] use histograms, proposing a histogram algebra for executing queries

on the histograms instead of on the data and estimating the quality of the approximate answers.

Authors in [1] revisit the database management systems architecture to combine sampling with speculative execution in the same engine. DICE [32] is a combination of a frontend query interface and distributed aggregation backend that enables interactive cube exploration. It allows the user to explore facets of the data cube, trading off accuracy for interactive response-times, by sampling the data. The front end presents an interface that allows for sampling-aware aggregations and encourages interaction via a faceted model.

Knowledge extraction based exploration. ADESCA [34] (Automated Data Exploration with Storytelling CApability), is a methodology to extract models and structures from data collections automatically. It supports the creation of a storyboard for automated data analytics, allowing the creation and visualization of data stories. ADESCA offers a guided step-by-step exploration, where the algorithm parameters are configured automatically and the results are graphically shown through different plots to be as understandable and user-friendly as possible. Helix [35] is a declarative machine learning system to optimize end-to-end workflow executions. READ [36] is a system for accelerating exploratory data analysis. READ decomposes the exploration process into components that can be independently specified and automated. Components can be defined, reused or extended using inference rules, planning logic, and reactive user interfaces and visualization. READ uses a formal specification of the analytic process for automated model space enumeration, workflow composition, deployment, and model validation and clustering. The READ engine reduces the time required for data exploration.

EPIQUE [37] is a generic topic evolution model enabling the specification and extraction of meaningful science evolution patterns from large document bibliographic archives. The goal of topic evolution networks is to track complex temporal changes by epoch-wise topic discovery and temporal similarity graphs aligning topics of different epochs. EPIQUE allows the audience to quickly and intuitively generate high-quality evolution graphs and explore them.

HARE [38] is a hybrid query engine that combines human and computer capabilities to enhance answers to SPARQL queries exploiting crowd knowledge. HARE relies on a completeness model to detect parts of a SPARQL query, potentially affected by missing values and should resort to the crowd. The query engine efficiently combines crowd answers and intermediate RDF results. HARE maintains knowledge bases that capture the knowledge col-

lected from the crowd, which is opportunistically exploited to discern whether the crowd is likely to solve a question accurately.

Visualisation based exploration. Data visualization is the most common tool for exploring and extracting insights from datasets, especially novice data scientists. Visual data exploration remains a tedious process of trial-and-error. Thus existing systems propose strategies to make the process user-friendly. DataWrapper, Datapine or ZenVisage [39] propose solutions for automating data exploration through graphic intuitive strategies with traditional visualization services, where the user explore the distribution of attribute values, choosing different types of charts. ZenVisage is based on ZQL ("zee-quel") for specifying the desired visual patterns, drawing from use-cases in domains like biology, mechanical engineering, climate science, and commerce.

In addition to the graphic analysis of the attributes, systems like AnswerMiner, DIVE [42] and NCSS[4] also analyse data. AnswerMiner allows to build decision trees on data. DIVE [42] allows to perform the Analysis of Variance (ANOVA[5]) test. NCSS allows to perform ANOVA test, multivariate analysis and a very simplified cluster analysis.

Authors in [43] present a vision of next-generation visual analytics services that seamlessly provide within an uninterrupted data analysis cycle (i) visual and interactive data exploration, (ii) suggest relevant data to enrich visualizations, (iii) facilitate the integration and cleaning of data.

Analytics platforms such as IBM's Watson AnalyticsTM collect metadata about their use, including user queries on uploaded datasets. The analysis of this metadata may be used for improving query recommendation and automatic data visualization. VIQS (Visual Interactive Query Semantics) [40] is a system that extracts query semantics from query logs over multiple datasets and allows users to explore underlying patterns visually.

Authors in [41] developed GraphTQL, a visual query system allowing end-users to formulate a query by transforming the diagrammatic representation of the schema graph. GraphTQL offers a high level of abstraction while allowing for feedback during the query formulation.

[4] https://www.ncss.com/

[5] Analysis of variance (ANOVA) is a collection of statistical models and their associated estimation procedures (such as the "variation" among and between groups) used to analyze the differences among means https://en.wikipedia.org/wiki/Analysis_of_variance.

2.1.3 Human-in-the-loop based data exploration

Exploratory Data Analysis is a human-intensive process involving data management, analytic flow deployment and model creation, and data visualization and interpretation. Thus, when designing a database exploration tool, a fundamental aspect to bear in mind is the relevance of an answer for a specific user. One of the most significant issues in exploration systems is that the system provides no explanations of the results or system choices. Nor does the system trigger input from the user, for example, by asking the user to provide more information. Data exploration can be modelled as a conversation between the user and the exploration system, defined as a multi-step, non-linear process with imprecise end-goals. The user provides some initial hints about her interests, and the system suggests "potentially interesting perspectives" over the data that may help refine the search. Table 2.3 summarises and illustrates human in the loop based data exploration methods.

Table 2.3: Human in the loop based exploration methods.

3. Human in the Loop Based Data Exploration	
3.1 Query similarity	Structured data (multiple tables, join and selection conditions, projections,possible groupings and aggregations) [45–48], find essential queries [49], query similarity [50, 51], [52]
3.2 Query recommendation	Top-n queries [5], cold/warm start [53, 54], previous query based [55], QuErie [46, 56], YMALDB [6], DIVE [42], ReDRIVE [50], [45, 57], [16]
3.3 Exemplar	Top-down/bottom-up web-based visual interactive [58], query by example [7, 8], exploration by explanation [59], T-EBAnO [60], [61]

Authors in [16] model the exploratory conversation as a lattice of nodes. Each node is a view over the database described intensionally as a conjunctive query and therefore represents a subset of database objects (tuples, in our relational case). Using this model, the user can have a precise idea of what she is looking for, or the system can suggest some initial relevant features that correspond to a sample of attributes' values. The notion of relevance is based on the frequency distribution of attributes in the view lattice.

Authors in [33] and [16] assist users [43, 44] in the exploration of transactional databases in an interactive way.

Exploration by explanation [59] enables a conversational setting, where the system can (a) ask clarifications and (b) explain results in natural language. This interaction assumes that the system can analyse and understand user requests and generate its answers or questions in natural language.

The work in [60] proposes T-EBAnO (Text-Explaining BlAck-box mOdels), a new engine able to provide both prediction-local and model-global interpretable explanations in the context of NLP analytics tasks that exploit

black-box and deep-learning models. This explanation framework allows users to understand why and how a prediction is made, allowing them to trust the model's outcomes consciously.

Zuo et al. [61] propose a scalable engine for the dynamic feature exploration over Time series stream, which is based on the interpretable Shapelet features and an explainable Shapelet extraction process. Shapelet is a time series subsequence which is particularly representative of a class. The explainability of Shapelet Extraction process is ensured even to non-expert, which offers us an option to further explore the streaming context conserving the current goodness.

Query similarity When queries are structured (including multiple tables, join and selection conditions, projections, and possible groupings and aggregations), the form of the query can be used to define query similarity [45–48] and then be utilized to find similar sets of tables and columns. Query similarity can also be used to identify essential queries [49]. In addition to using a syntactic notion of query similarity, some of these approaches also measure how related the query results of the recommended query would be to a given query [50, 51].

In [52], the authors propose an approach for rewriting aggregate queries using a cache containing the answers to previously asked queries. Their approach relies on a semantic matrix and is generic enough to consider queries involving selection, projection and usual aggregation functions. An essential feature of the approach is that it can generate several rewritings for a given query.

Query recommendation Interactive exploration is becoming increasingly popular in the scientific community. This popularity has tremendously increased the need for a recommendation system and various tools as the user cannot explore data. This requirement motivates the emergence of query recommendation systems that suggest top-N query recommendations according to initial requirements. Thus, they assist users in improving search quality when users are not satisfied with the results of an initial input query.

Exploration by recommendation guides the user in what possible actions to perform or data to look at next. Both cold-start (where the user has not given any input) and warm-start settings (where the user has asked one or more queries but may not know what to do next) can be considered. In the cold start, the goal is to show a set of examples or starter queries that the users could use to get some initial answers from the dataset (e.g. [53]). In

the warm start case, the system can leverage the user's interactions (queries) to show possible following queries (e.g., [54]).

In structured databases (including data warehouses), several query recommendation systems have been proposed [45, 57] to help users form queries or find similar or related data. A combination of syntactic query similarity and instance-based similarity (see below) can be helpful for query recommendation over a single (coherent) database where all users use the same schema.

The paper [55] proposes a method that suggests a list of queries related to the user input query. The related queries are based on previously issued queries and can be issued to the search engine to tune or redirect the search process. The method is based on clustering that uses the content of historical preferences of users registered in the search engine's query log.

The problem of generating recommendations has been broadly addressed in the web context. Query Steering [4] relies on the user's feedback to provide suggestions. QueRIE [5] is a Query Recommendation Framework that uses recommendation techniques and algorithms to assist technical and non-technical users to express and redefine queries executed on Sky Server. This framework generates top N personalized query items using an item to item recommendation algorithms[6]. In the project, QueRIE (Query Recommendations for Interactive database Exploration) [46, 56] the idea is to track the querying behavior of each user, identify which parts of the database may be of interest for the corresponding data analysis task, and recommend queries that retrieve relevant data. The principle is to focus on user-based collaborative filtering: *if user A has similar querying behavior to user B, then they are likely interested in the same data. Hence, the queries of "user B" can guide user A.* Based on this premise, the similarity between users is expressed as the similarity between the fragments of their queries or the data that they retrieve.

The YMALDB framework [6] is a database system enhanced with a recommendation functionality. It allows users to discover highly correlated and similar tuples to the original query's results. For a given user query, YMALDB computes additional results, called Ymal (i.e., "You May Also Like") results, that are highly related to their initial query results. Such results are computed using the most interesting sets of attribute values, called faSets, that appear either in the initial query results or in the results

[6]Item-item collaborative filtering is a form of collaborative filtering for recommender systems based on the similarity between items calculated using people's ratings of those items. It was invented and used by Amazon.com in 1998. It was first published in an academic conference in [62].

of an appropriately expanded one. The degree of interest of a faSet is given by its frequency both in the query result and in the database.

DIVE [42] combines recommendation strategies with a point-and-click manual specification to support data model inference, visualization, statistical analysis, and storytelling capabilities. The algorithm proposed in [28] clusters queries according to four notions: the context of the query, common clicked URLs between queries, similar strings between the queries and the distance of the clicked documents in some pre-defined hierarchy. Another recommendation method is based on association rules to discover related queries, where queries represent items in traditional association rules [63]. The query log file is considered a collection of transactions representing a session where the user submits all related queries at a specific time.

ReDRIVE (Result-Driven Database Exploration through Recommendations) [50] assists users in database exploration by recommending additional items that are highly related to the items in the result of their original query. Such items are computed based on the most interesting sets of attribute values (or faSets) in the original user query result. The interestingness of a faSet is defined based on its frequency both in the query result and in the database instance (i.e., it expresses how unexpected it is to see this faSet in the result).

To support real-time exploration with user feedback [64] introduces a new class of continuous top-k queries featuring complex dynamic scores. The main problem they address is the search of queries that need to update their result lists when new information items and user events affecting their scores arrive. The authors propose three general categories of in-memory indexes and heuristic-based variations for storing the query candidates and retrieving the result updates triggered by events to accommodate high arrival rates of both items and events. Using the proposed model, they demonstrate MeowsReader [65] a Real-Time Ranking and Filtering of News with Generalized Continuous Top-k Queries. In MeowsReader, users define their interest by issuing text queries for continuously receiving the best matching results in an alert-like environment.

Exemplar based exploration. Example-based methods are pertinent when the user is aware of some items that match (at least partially) her interest. Existing by-example methods are used to infer SQL queries from some example tuples, identify rules to align two databases, and find distinct entities that represent the same object. For documents, example-based methods identify the topics of interest from the example to return other unexpected documents. In a graph, starting from a structure, they navigate

the nodes and edges conveniently and show relevant areas and similar objects or structures [14].

In general, query-by-example techniques [7, 8] receive as various input examples of expected results to prevent the user from formulating a concrete and precise query. Finally, [9] proposes interactive and efficient navigation across large data spaces based on data cluster information.

Works applied to relational example-based data exploration start from tuples, tuple pairs, or example datasets and explore data [14]. Salient techniques for relational example-based data exploration include families of reverse engineering queries. Approaches can be classified according to whether the queries return approximately, or precisely the set of example tuples. They can also be classified according to the different query operators considered on the reverse engineering process. Reverse engineering querying can be further classified into exact (one step, interactive) and approximate (minimal, top-K).

Exploration by Example [59] is an operator that takes a set of examples and explores its different facets, filters them, finds similar/dissimilar sets, finds overlapping sets, joins them with other sets, finds a superset, etc. Additionally, by-example operators can be combined with by-analytics to find sets that are similar/dissimilar concerning some value distributions.

Authors in [58] develop a web-based visual interactive exploration system of network data. The approach combines two well-known network exploration strategies: top-down and bottom-up. The top-down strategy provides an overview of network structures and guides analysts to local details by filtering or querying. The bottom-up strategy shows local details upon the request of analysts and supports network exploration by following nodes and edges of interest. Both strategies are integrated through an "examplar" based strategy that lets users quickly analyze and compare similar structures in examining an extensive network.

User expectations relevance remains a subjective factor, the immediately-related research challenge is to capture users' interests to issue the most relevant answer. Therefore, user studies are crucial to classify different notions of relevance and devise strategies to customize the exploration system response to the user behavior. Achieving this classification requires identifying real scenarios and involving user groups to understand how they access data and what they perceive as the data exploration needs to be addressed by current systems.

An exploratory interface should support appealing, synthetic visualization of the query answers. It should also highlight the relevant properties of current and past query answers to let users get insight into the data and decide how

and to which extent to explore them.

Discussion

Research has been devoted to automating the data exploration process. Data exploration approaches have been proposed in different contexts and have not been integrated into the same environment. Query Expansion methods involve the reformulation of queries by adding new terms to the initial user query, and Query Recommendation methods propose the nearest queries to the initial query, taking as an input a set of queries.

Therefore, there is room for proposing approaches for each of them, defining rules on how they can be combined within data exploration pipelines and integrating them to provide a whole data exploration environment. The definition of a data exploration environment concerns the objective of our work.

2.2 Exploration systems and tools

Data exploration systems apply machine learning techniques, multivariate statistical methods, information theory, and database theory to databases to identify significant relationships among the data and summarize information. Applying data exploration systems should create a better understanding of the structure of the data and a perspective of the data, enabling an analyst to form hypotheses for interpreting the data. According to [59], a full-fledged data exploration system must combine different access modalities with a powerful concept of guiding the user in the exploration process by being reactive and anticipative both for data discovery and for data linking.

2.2.1 Data exploration systems

The DBMS applications are becoming increasingly popular in the scientific community to support the interactive exploration of massive data. M. L. Kersten et al. [24] have compiled methods to explore datasets querying. These query systems provide a broader (i.e. less precise but wider scope) approach, discarding exactness and completeness for speed and a more global vision of the data.

One-minute DB kernels focus on executing a query within a strict time limit instead of traditional databases, which focus on correctness and completeness. This kernel differs from conventional kernels by identifying and avoiding performance degradation points on the fly and answering part of the

query without changing the query focus. One-minute DB kernels sacrifice correctness and completeness for performance. This sacrifice helps data exploration allowing users to perform more or less accurate queries till he or she identifies the best query to answer a specific question.

INODE [59] is an end-to-end data exploration system - that leverages, on the one hand, Machine Learning and, on the other hand, semantics for Data Management (DM). INODE brings together different data management solutions to enable intelligent data exploration in an architecture consisting of the following components: (1) Data Modeling and Linking enables integration of both structured and unstructured data. (2) Integrated Query Processing enables efficient query processing across federated databases leveraging ontologies. (3) Data Access and Exploration enables guided data exploration in various modalities such as natural language, certain operators, or visually.

Précis [51] queries represent a novel way of accessing data, which combines ideas and techniques from the fields of databases and information retrieval. They are free-form, keyword-based queries on top of relational databases that generate entire multi-relation databases, which are logical subsets of the original ones. A logical subset contains items directly related to the given query keywords and items implicitly related to them to give the user much greater insight into the original data.

SnapToQuery [26] guides users through a query space by providing interactive feedback during the query specification process by "snapping" to the user's likely intended queries. These intended queries can be derived from prior query logs or from the data itself.

DBNav (Database navigator) [66] is a service that assists as a "tour guide" to facilitate interactive data exploration. DBNav is based on query steering, which refers to assisting a user to navigate through complex data space. DBNav steers the user towards interesting "trajectories" through the data while highlighting relevant features. Profiles of both user interest and application characteristics are key notions for DBNav. Profiles can be either supplied by the user or learned from interaction logs. DBNav proposes three steering patterns: manual, power, and auto steering.

DBXplorer [67] is a system that enables keyword search over SQL databases. Given a set of query keywords, DBXplorer returns all rows (either from single tables or by joining tables connected by foreign-key joins) such that each row contains all keywords. DBXplorer was implemented using a commercial relational database and web server, and allows users to interact via a browser front-end.

Finally, Bleau [9], Ziggy [68], and QPlain [69] show the potential of example-based approaches in escorting the user toward the wanted answers.

Such exploration tools have been introduced to assist the user in formulating queries with simple interfaces that assist the query formulation and understanding phase.

QPlain [69] uses provenance models [70] to generate additional explanations for the user. The provenance models are conjunctive queries with inequalities, which can be represented as provenance polynomials [71]. A query is treated as an expression with two operations in provenance polynomials: multiplication indicates joint tuples, and summation indicates alternative derivations. Such explanations materialize different results proposed to the user, who is engaged in a search-and-refine process.

Bleau [9] adopts a clustering approach over the possible tuples by partitioning the data along two dimensions: groups of mutually dependent attributes (columns) that represent a topic, and each group is partitioned internally using the attribute values in a tree-shaped visualization. The resulting trees are then shown to the user, who can easily navigate complex portions of the dataset.

2.2.2 Data wrangling environments

Data wrangling (i.e., data munging) is the process of transforming and mapping data from one "raw" data form into another format for making it more appropriate and valuable for a variety of downstream purposes such as analytics[7]. Data wrangling is a process including munging, visualization, aggregation, and training a statistical model.

Data wrangling has traditionally been performed manually (e.g. via spreadsheets such as Excel), tools like KNIME[8], or via scripts in languages such as Python or SQL and R. Visual data-wrangling systems make data wrangling accessible and more straightforward. They can include AI recommenders and programming by example facilities and program synthesis techniques to generate scalable dataflow code. Early prototypes of visual data wrangling tools include OpenRefine[9] and the Stanford/Berkeley Wrangler[10] research system that the latter evolved into Trifacta.

Trifacta[11], provides tools to explore, transform, and enrich raw data into clean and structured formats. It uses machine learning, data visualization, human-computer interaction, and parallel processing to let users work with

[7] https://en.wikipedia.org/wiki/Data_wrangling - accessed the 3rd July 2021
[8] https://www.knime.com - accessed the 3rd July 2021
[9] https://openrefine.org - accessed the 3rd July 2021
[10] http://vis.stanford.edu/wrangler/ - accessed the 3rd July 2021.
[11] https://www.trifacta.com - accessed the 12th June 2021

datasets. Tableau[12] provides tools to extract, store, and retrieve data from relational databases, online analytical processing cubes, cloud databases, and spreadsheets to generate graph-type data visualizations that can help explore data collections.

INDIANA [33] is an interactive system for assisting transactional databases exploration who are interested in gaining insights about a database through an interactive and incremental process. The system interactively provides the user with some data features that might be "interesting" from the statistical viewpoint, receiving feedback to refine the features provided to the user.

Ziggy [9] supports the user with statistics over a set of examples decomposed into views over groups of attributes in which the statistical distribution of the user's tuples is different from that of the rest of the data. The views are ranked using a difference score among the data and the examples and validated through conventional statistical tests, such as chi-square. Ziggy sustains high-dimensional data and needs little supervision from the user. Such visualizations are helpful for navigating the data, but the use of examples is still limited to pure search-and-click approaches or complex explanations which the user does not directly control.

2.3 Data exploration classification

According to Wikipedia, data exploration is an approach *whereby a data analyst uses visual exploration to understand what is in a dataset and the characteristics of the data (e.g., size or amount of data, the completeness of the data, correctness of the data, possible relationships amongst data elements), rather than through traditional data management systems.* In the context of this work, data exploration is a process where users go through a large dataset in an unstructured way to uncover initial patterns, characteristics, and points of interest. Data exploration can use a combination of manual methods and automated tools like data visualizations, charts, and initial reports. Thus, data exploration calls for combining different querying and processing methods and strategies proposed in diverse domains: databases, data mining, information retrieval, data analytics, data science, data profiling, etc. Every discipline proposes a definition of data exploration, even if in general, proposals define it as a process. To acquire a better understanding of data exploration, we propose a classification grid to study existing work that resulted from our systematic mapping (see Figure 2.1).

According to a systematic review we performed (the quantitative study

[12] https://www.tableau.com - accessed the 12th June 2021

F1. Types of queries	**D1.1. Structure exploration**	**G1.1.1. Static structural** *g1.1.1-a Relational* *g1.1.1-b Geo/Spatio-temporal*
		G1.1.2. Flexible structural *g1.1.2-a Navigational*
	D1.2. Content exploration	**G1.2.1. Keyword search** *g1.2.1-a Basic* *g1.2.1-b Context* *g1.2.1-c Phrase* *g1.2.1-d Proximity* *g1.2.1-e Boolean*
		G1.2.2. Pattern matching
		G1.2.3. Query by example *g1.2.3-a Query suggestion* *g1.2.3-b Reverse engineering* *g1.2.3-c Entity extraction by example text* *g1.2.3-d Web table completion with examples* *g1.2.3-e Search by example* *g1.2.3-f Community based node retrieval* *g1.2.3-g Entity search* *g1.2.3-h Path and SparQL queries* *g1.2.3-i Graph structures by example*
	D1.3. Quantitative dataset profile	**G1.3.1. Descriptive**
		G1.3.2. Predictive
F2. Type of process done by a human	**D2.1. Giving feedback**	
	D2.2. Visual Exploration	
	D2.3. Specification of Parameters	**G2.3.1. Group Recommendation** *g2.3.1-a Consensus functions* g2.3.1-b *Group preferences (average, last misery)*
		G2.3.2. Group Disagreement *g2.3.2-a Average pairwise* g2.3.2-b *Disagreement variance*
F3. Data exploration techniques	**D3.1. Queries based exploration**	**G3.1.1. Query expansion** **G3.1.2. Queries as answers** **G3.1.3. Multi-scale queries**
	D3.2. Model Based Exploration	**G3.2.1. Statistics** **G3.2.2. Knowledge** **G3.2.3. Visualization**
	D3.3. Human-in-the-loop	**G3.3.1. Query similarity** **G3.3.2. Recommendation** **G3.3.3. Exemplar**
	D3.4. Tools and Systems	
F4. Data sets	**D4.1. Provider**	**G4.1.1. Newspaper** **G4.1.2. Social network** **G4.1.3. City infrastructure**
	D4.2. Type	**G4.2.1. Image/video** **G4.2.2. Phone call** **G4.2.3. Geo/spatio temporal** **G4.2.4. Transport**
	D4.3. Topic	**G4.3.1. Disaster** **G4.3.2. Human mobility** **G4.3.3. Transport**

Figure 2.1: Querying techniques for exploring datasets

of the systematic review can be seen in Appendix A.1), we propose a classification of existing data exploration techniques and methods and associated issues like query and data collection types. The classification consists of facets representing data exploration aspects and dimensions that denote the concepts that define each facet. Some dimensions in our analysis can further represent specific strategies that we name groups. As shown in Figure 2.1, the proposed facets:

- (F_1) the type of queries addressed by existing work and the type of algorithms used by different techniques for exploring data collections;
- (F_2) the exploration tasks done with human intervention;
- (F_3) data exploration techniques and systems conceived for understanding raw datasets content;
- (F_4) the knowledge domain of data collections and data types.

As shown in the figure and as we studied the literature more thoroughly, dimensions tagged by DX.X associated to facets were further decomposed in finer grain values tagged by GX.X.X and these groups into subgroups gX.X.X-i with i ∈ {a, b, ...z } (see Figure 2.1). The following lines describe the facets and (sub) dimensions, then the next sections summarise the main work and proposed approaches addressing the dimensions of each facet.

2.3.1 Types of data exploration queries

Since exploration can put different types of queries in action, *facet* F_1 classifies the types of queries defined and their evaluation process in different works that explore datasets. Data exploration queries guiding the exploration task assume that the user has different degrees of preliminary knowledge about the content of a collection. The spectrum goes from "classic" keyword and relational queries evaluated over more or less curated datasets, to data processing operations on raw datasets (e.g., descriptive statistics, machine learning). In this spectrum, these types represent families of queries that include conjunction/disjunction operators, relational, filtering, statistical, aggregation and clustering operations. *Facet* F_1 classifies queries in terms of the way they are expressed and whether they refer to the structure (d1.1), the content of data collections (d1.1) and quantitative properties (d1.3).

2.3.1.1 Structure exploration queries

Queries expressed with respect to the structure of a data collection to explore (D.1.1) rely on both (i) the knowledge of a (semi) structured content to explore;

and (ii) complete knowledge of the content structure (i.e., schema, document (partial) structure). Some query types are precise like *static structural queries*. For instance, *relational queries* clearly specify the item they want to explore (e.g., name of the column in tabular data collections). Other query types are *flexible structural queries* like *navigational queries*. For instance regular or FLWOR expressions [72] defined to explore (semi)structured documents' collections, or matching/path queries when they refer to graph structures.

Temporal and spatial queries are used to explore data collections containing time series or spatial attributes that determine their content. The items of these data collections have an inherent order determined by time stamps and/or locations. Therefore, exploration can be guided by queries that enable the specification of intervals/window, points, or operations that can connect or establish some relation among series of points (e.g. connected points).

2.3.1.2 Content exploration queries

Content exploration queries can be expressed to look up content disregarding the structure of the data collection (D1.2). The challenge is to represent the content that a user is searching for by giving examples of her target or describing some pattern or characteristic to match. Keyword, pattern matching and query-by-example are the most prominent types used for exploring content. The query expression is closely related to the format of the content (e.g., textual, multimedia).

Keyword search queries (G1.2.1). The simplicity of keyword search as a querying paradigm offers compelling values for data exploration because it does not require a priori knowledge of the schema.

- *Keyword queries* (g1.2.1-a) are the most common expressions used for exploring textual data collections. A keyword is a word which occurs in a text more often than we would expect to occur by chance alone. Keywords are calculated by carrying out a statistical test (e.g., loglinear or chi-squared) which compares the word frequencies in a text against their expected frequencies derived in a much larger corpus, which acts as a reference for general language use. The objective of keyword search is to generate, with good precision and recall, a large number of terms that are highly relevant yet non-obvious to the given input keyword.
- *Keyword context queries* (g1.2.1-b) complement single keyword queries with the ability to search words in a given context (i.e., words that

appear near each other may be likely to be more relevant). They can be derived into *phrase queries* and *proximity queries*.

- A *phrase query* (g1.2.1-c) is a sequence of single-word queries, where separators between words do not need to be the same in target texts and the query expression (e.g., two spaces versus one space).
- A *proximity query* (g1.2.1-d) consists of a sequence of single words or phrases with an associated maximum allowed distance between them.
- *Boolean queries* (g1.2.1-e) have a syntax composes of atoms (i.e., basic queries) that retrieve documents and Boolean operators which work on their operands (i.e., which are documents' sets), and deliver documents' sets. Since operators can be composed over the results of other operators a query syntax is defined, where leaves correspond to the basic queries and internal nodes to the operators. The query syntax tree operates on an algebra over sets of documents and the final answer of the query is also a set of documents. The most commonly used operators are OR, AND, BUT ($e_1 BUT e_2$ selects all documents which satisfy e_1 but not e_2).

Pattern matching (G1.2.2) are query formulations based on the concept of pattern, which allow retrieving pieces of text with some property. They are popular expressions for linguistics, text statistics and data extraction. A pattern is a set of syntactic features that must be found in a text segment. Those segments satisfying the pattern specification are said to match the pattern. Different solutions and systems allow specifying some types of patterns, which range from very simple (e.g., a string of characters) to rather complex (e.g., regular expressions). In textual collections patterns can refer to word prefixes, suffixes, substrings and ranges[13]. Pattern matching can be applied to other types of content format, like multimedia and graphs. The pattern forms and expression change and of course the search techniques too.

Query-by-example is an intuitive way to explore data, so many techniques are applying it to data exploration (G1.2.3). We mainly identify "query-by-example" techniques, particularly when the knowledge about the datasets' content is too weak. In relational databases it is a graphical query language, using visual tables where the user enters commands, example elements and conditions. In the context of information retrieval, the user can submit a document, or several documents, and ask for "similar" documents to be

[13] A pair of strings which matches any word that lexicographically lies between them.

retrieved from a document database. Other query-by-example approaches include reverse engineering querying and queries like query morphing. Other techniques aligned to query-by-example include entity extraction by example text[14], Web table completion with examples and search by example. Finally, in graph data exploration there are techniques like community based node retrieval, entity search, path and SparQL queries, and graph structures by example.

2.3.1.3 Quantitative dataset profile queries

Quantitative queries (D1.3) focus on the mathematical profile of the content of explored data collections.

Descriptive queries (G1.3.1) explore the structure and possible metadata describing data collections like size, format, structure, topic, textual description, versions, release dates, quality measures, etc. This type of query is keyword-based and is often implemented by data labs like Kaggle or data providers like Wikipedia, Stack Overflow, Social networks, etc.

Authors in [73] consider the continuous preference queries (i.e., preferences queries over data streams) in which user preferences are expressed by if-then rules composed by predicates over attributes. The cp-queries allow several types of applications return more customized answers to users. The work proposes an efficient approach for the evaluation of continuous cp-queries. Their exhaustive set of experiments show a considerable better performance of proposed technique with respect to state-of-art algorithms. The textual content of data collections can be described statistically, counting frequent terms or computing the percentage of every term within one or several documents.

Statistical queries (G1.3.2) explore the data collections content willing to acquire a quantitative insight of the content through well-known measures (values distribution, min, max, std, percentage of null values, etc.). In the case of tabular and semistructured documents collections, statistical exploration produces measures for every attribute of a table and every table composing a collection. In the case of non-structured data collections like images, the statistical measures can be related to the percentage of the colours in the images, the number of salient objects, etc. Queries ask about

[14]Entity extraction is a text analysis technique that uses Natural Language Processing (NLP) to automatically pull out specific data from unstructured text, and classifies it according to predefined categories such as phone numbers, monetary values, or dates.

these statistical measures, for example, the distribution of the values of a given keyword within the textual content, the most popular colours within a set of images, and the collections with the most missing values.

Predictive queries (G1.3.3) define pipelines that include data cleaning and wrangling tasks, statistical analysis and models (e.g. regression) to predict the evolution of some item of the data collection. For example, for a data collection gathering, energy consumption readings within a time interval, this type of query can estimate readings in "future" intervals.

Depending on the domain, works propose algorithms rather than operators (like in relational contexts) to process datasets and to discover and derive a precise statistical understanding of their content. Algorithms sometimes depend on the type of data structures used for representing data. For example, there are algorithms for processing graphs (centrality, pathfinding, etc.) or querying tables (selection, projection, etc.). Many works use well-known heuristics, data mining, machine learning, artificial intelligence algorithms for processing datasets, and insight into their content.

2.3.2 Human guided exploration

The vision of data exploration is that it should be a human-guided process (F_2). Thus, data exploration is generally expressed as a process that lets the user successively approach with some degree of understanding of the data collection content. The process can be done "top-down", starting with a query that expresses a starting "guess" related to the content or structure of the data collection items. Therefore, exploration is designed as a conversation that provides data collections' samples and/or alternative queries to answer a query (D2.1). Queries are refined successively until the user stops the exploration process or until some assessment score is achieved. The process can also be done "bottom-up" by generating a content representation, such as a vocabulary or frequent terms extraction in textual collections, or visual graphics, clustering and videos resume. The user can then explore these synthesized views of the content playing with granularities, zoom in/out operations, etc. The user can intervene in the whole process, including choosing graphics or methods and algorithms or just in some phases (D2.2).

2.3.2.1 Group recommendation and consensus functions.

We have studied techniques where humans intervene to adjust and guide the process of receiving information (d2.3). Particularly, we first studied works on group recommendation (G2.3-1), consensus functions (g2.3-1a),

group preference (g2.3-1b) and group disagreement (G2.3-2). This study address objectives like designing consensus functions that aggregate individual group members' preferences to reflect the overall group's preference for each item [24, 74, 75] or disagreement about an item [76]. Consensus functions can be applied within a data exploration process given where a user can agree and disagree about the proposed queries; the system can recommend queries according to given constraints that can be interpreted as preferences. The next lines describe these techniques.

Group recommendation (G2.3-1) refers to finding the best items that a set of users will appreciate together. It is an active research area as exemplified by numerous publications [24, 74, 75]. The main focus of existing work in group recommendation is the design of appropriate consensus functions that aggregate individual group members' preferences to reflect the overall group's preference for each item. A variety of consensus functions have been used, ranging from majority voting to most minor misery.

- *Consensus functions* in a group, members may not always have the same preferences for items, and a consensus function needs to aggregate user-item preferences into a single group's preference for an item. Intuitively, there are two main aspects in a consensus function [76] (g2.3.1-a). First, the preference of a group for an item needs to reflect the degree to which all group members prefer the item. The more group members prefer an item, the higher its group preference. Second, the group preference needs to capture the level at which members disagree or agree. All other conditions being equal, an item that draws high agreement should have a higher score than an item with a lower overall group agreement. We call the first aspect group preference and the second aspect group disagreement. We revisit the definitions that were introduced in [24].
- Group preference [24] (g2.3.1-b) the preference of an item i by a group G, denoted gpref (G,i), is an aggregation over the preferences of each group member. Two commonly used aggregation strategies are:
  - Average preference computes the group preference for an item as the average of individual group members' preferences for that item.
    $$\frac{1}{|G|}\sum(pref(u,i)) \tag{2.1}$$
  - Least-Misery preference computes the group preference for an item as the minimum among individual group members' preferences

for that item.

$$Min(pref(u,i))) \tag{2.2}$$

Group disagreement (G2.3.2) [24]: The disagreement of a group G over an item i , denoted $dis(G,i)$, reflects the degree of consensus in the user-item preferences for i among group members. Two most common disagreement computation methods are:

- Average Pair-wise Disagreements computes the group preference for an item as the combination of its average and its pair-wise disagreement between individual group members' preferences (g2.3.2-a).

$$\frac{2}{|G|\,(|G|-1|))}\sum(|pref(u,i)-pref(v,i)|) \tag{2.3}$$

  where $u \neq v$ and $u, v \epsilon G$.

- Disagreements Variance is the mean of all the individual preferences for item i, which computes the mathematical variance of the preferences for the item among group members (g2.3.2-b).

$$\frac{1}{|G|}\sum(pref(u,i)-mean(i))^2 \tag{2.4}$$

  where mean(i) is the mean of all the individual preferences for item i.

The average pair-wise disagreement function computes the average pair-wise differences in preferences for the item among group members, while the variance disagreement function computes the mathematical variance of the preferences for the item among group members. Intuitively, the closer the preferences for i between users u and v, the lower their disagreement for i.

Consensus functions can be applied within a data exploration process given where a user can agree and disagree about the proposed queries; the system can recommend queries according to given constraints that can be interpreted as preferences. In order to understand these techniques, we applied them, implementing a GroupTravel framework that generates customized travel packages (TPs) for a group of individuals [77].

2.3.3 Data exploration techniques

Exploratory data analysis (EDA) is the process of discovering salient characteristics of a dataset or finding data-driven insights in the corresponding

domain. Challenges of data exploration are data complexity, ambiguous query intent, continuous actions (users articulate queries using gestures, treating query specification as a continuous process) and consideration of feedback. As said before, we note that data exploration is a loop that obtains approximated results and the techniques are specialised according to the type of data models (relational, graph, semi-structured, text, multimedia).

According to our classification, facet F_3 considers dimensions that represent three families of exploration techniques. M. L. Kersten et al. [24] have compiled five methods to explore datasets querying: one-minute DB kernels, multi-scale queries, result set post-processing and query morphing and queries-as-answers. We have further identified other techniques and propose the following classification.

Queries based data exploration characterised by query expansion primarily explored by information retrieval solutions (D3.1). Other techniques focus completely in guiding the user to express queries that will potentially produce "good quality" results. The queries-as-answers family represents techniques that given a query produce as answers new queries (D3.2). Human in the loop exploration techniques (D3.3) are interactive strategies that intend to guide the expression of queries, the way they are refined and the way results are presented visualised to give insight into data content.

Finally, data exploration systems & environments (D3.4) are tailored for exploring data incrementally and adaptatively. These systems revisit fundamental characteristics of existing systems like the notion of results completeness and correctness promoted by traditional databases, splitting queries execution on different fragments of a database, precision of queries, and one-shot computation of query results. These query systems provide a broader (i.e., less precise but broader scope) approach, discarding exactness and completeness for speed and a more global vision of the data.

2.3.4 Types of data collections to explore

Facet F_4 classifies the type of datasets that are accessible through different providers, for example, Kaggle, Wikipedia, Social Networks, etc. In general, data providers encourage data collections owners to share their data in an accessible, non-proprietary formats that can be better supported by transformation functions and that are also easier to work regardless of the tools.

Datasets content is often textual and with different rawness degree (newspapers, micro-texts from social networks) and already processed content using NLP (Natural Language Processing) techniques and represented as

graphs or tables (e.g., CSV). Other datasets are built by collecting observations monitored using, for example, IoT infrastructures. These datasets contain records of measures or even video or images.

Many data lab platforms are not simple repositories. Each dataset is a community where users can discuss data, discover public code and techniques, and create their own projects.

Discussion

We have the following remarks about what we have studied in state of the art. Data exploration strategies are mostly ad-hoc, implemented in an artisanal manner, and partially human-guided. Machine learning, analytics, and querying techniques like query-by-example, queries-as-answers are complementary. We observed that no existing system integrates them so that data scientists can develop exploration pipelines that can thoroughly understand data and its analytics potential.

These data exploration approaches have been proposed in different contexts and have not been integrated into the same environment. Therefore, there is room for proposing approaches for each of them, defining rules on how they can be combined within data exploration pipelines and integrating them to provide a whole data exploration environment.

2.4 Conclusions

The evolution in querying, information retrieval, and human-computer interaction has shifted interest from the traditional query-response paradigm to existing human-guided data exploration systems. A traditional query-response system is limited to ”facts finding” (e.g., find hotels in the city centre). Modern exploration driven processes call to rethink data querying processes gathering, understanding and using insightful users' information [1] that act as relevance criteria for guiding the query search iterations [2, 3]. Existing works propose different exploration strategies and approaches [4–9].

Existing query rewriting techniques are oriented towards information retrieval requirements and not towards exploration. Queries are rewritten (expanded, morphed into one or several queries) to increase the possibility of matching as much documents as possible, but not to propose queries alternatives associated to samples where the user can have insight into the scope of documents that are available and can potentially fulfil target requirements. We have addressed this aspect by proposing and adjusting

the principle of existing query morphing, queries as answers and query by example techniques to tackle textual document exploration (see Chapter 4).

Only a few described systems and approaches have considered creating a workflow that combines different data exploration techniques within interactive exploration processes. In this work, we study and address the problem of assisting data consumers to effectively and interactivity explore the data collections for computing query sets that can better exploit raw data collections and discover the type of analysis and questions that such collections can answer. Therefore, we have proposed HILDEX, a human-guided data exploration environment that combines exploration operations into workflows that result in different types of factual and analytic queries used to explore datasets content (see Chapter 3). The idea is to include the interaction with the user to propose and adjust the type of patterns and questions that can be answered using given data collections.

CHAPTER 3

Query Exploration Algorithms

This chapter describes in detail the data exploration algorithms we propose. Namely, query morphing, queries-as-answers and query-by-example adapted for processing Boolean keyword queries asked on top of textual data collections.

The chapter is organized as follows. Section 3.1 introduces preliminary concepts and assumptions used in the proposed data exploration algorithms. Section 3.2, Section 3.3 and Section 3.4 describe respectively the proposed query morphing, queries-as-answers, and query-by-example algorithms. The sections describe the general principle of the corresponding algorithm and an example that illustrates its use. Section 3.5 describes a set of experiments conducted for validating the algorithms and evaluating their performance in terms of metrics. It also discusses and compares results. Finally, Section 3.6 concludes the chapter insisting on the properties of the algorithms and possible enhancement opportunities.

3.1 Preliminary concepts

The algorithms proposed in this work target the exploration of textual documents collections using Boolean keyword queries. The content of textual documents and queries (which are considered documents) is processed using information retrieval techniques that include pre-processing the content to clean it by filtering stop words and performing stemming and lemmatization. The result is a cleaned documents' set, where a bag of terms represents each document's content.

3.1.1 Indexing textual documents content

Indexing is the process of scanning the text of all the documents in a collection to build a list of search terms (often called an index). The process makes an entry in an index (see Inverted index below) for each term found in a document, and notes its relative position within the document. Some indexing techniques use language-specific stemming on the terms. For example, the words "drives", "drove", and "driven" are recorded in the index under the single concept term "drive".

Frequency Matrix is a mathematical matrix that describes the frequency of terms that occur in a collection of documents. *"In a frequency matrix, rows correspond to documents in the collection and columns correspond to terms. Each ij cell, is the number of times word j occurs in document i. As such, each row is a vector of term counts that represents the content of the document corresponding to that row".*[1]

In this work, the frequency matrix is built using pseudo-relevance feedback (or blind feedback) [78]. The idea behind the pseudo-relevance feedback relies on the assumption that the top k ranked documents are more relevant to the query, and most of the frequent terms in the pseudo-relevance documents are useful for query expansion[2]. Thereby, the frequency matrix is built as as follows:

1. Consider top-k documents retrieved by an initial query as relevant results.
2. Select top-m terms from these documents using tf-idf weights[3] and append them in the frequency matrix.

Inverted Index is a database index storing a mapping from content, such as words or numbers, to its locations in a document or a set of documents. It allows fast full-text searches. It is used to solve questions of the type *find the documents where word X occurs.*

Word embedding is a learned representation for text where words that have the same meaning have a similar representation. Each word is mapped

[1] https://en.wikipedia.org/wiki/Document-term_matrix - Accessed the 10th. July 2021.

[2] This method is widely used in many information retrieval tasks [79, 80].

[3] tf-idf is a weighting scheme that assigns each term in a document a weight based on its term frequency (tf) and inverse document frequency (idf). The terms with higher weight scores are considered to be more important https://www.geeksforgeeks.org/tf-idf-model-for-page-ranking/ - Accessed 10th July 2021.

to one vector that encodes the meaning of the word such that the words that are closer in the vector space are expected to be similar in meaning. Word representations are typically learned by modeling local contexts of words, assuming that words sharing similar surrounding words are semantically close (i.e., local embedding). Global contexts, referring to the broader semantic units, such as the document or paragraph where the word appears, can capture different aspects of word semantics and complement local contexts (i.e., global embedding).

Vector Space Model (VSM) represents documents as vectors in a vector space. In this space, points that are close are considered semantically similar, whereas points that are far away are semantically distant [81]. Search engines commonly use VSMs to determine the similarity between a query and a document [82].

A pre-trained embedding model is a type of vector space model to transform queries and documents, respectively, into a query vector and a document vector. The principle is based on the assumption that a document (query) content can be represented as a bag of words $\{W_1, W_2, ..., W_n\}$.

Vectorization process. A document (query) vector is computed by summing the word-vectors in the bag of words and then dividing the vector summation by the number of words in the bag (see figure 3.1). Since embedding is additive [83], the document-vector and query-vector can be considered to capture the semantics of all words in the document and the query, respectively.

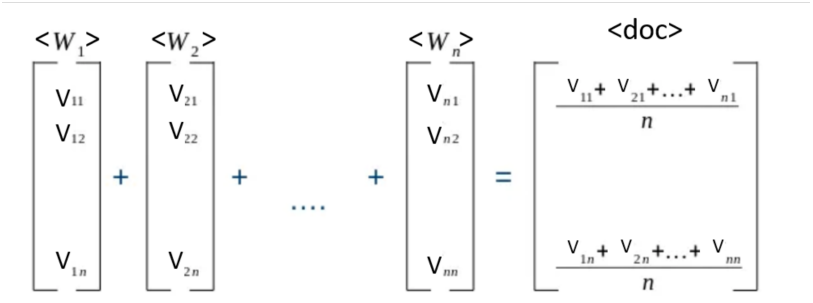

Figure 3.1: Document-vector representation. $< W_i >$ shows the vector of term W_i in a document and $< doc >$ shows the vector of that document.

Similarity between to documents is given by the cosine similarity, a metric that quantifies the similarity between two vectors. It mathematically calculates the cosine of the angle in a multi-dimensional space between two n-dimensional vectors. For the given two vectors, A and B, the cosine similarity can be obtained through the following formula:

$$Similarity(A, B) = cos(\theta) = \frac{A.B}{||A||\,||B||} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \tag{3.1}$$

where A_i and B_i are components of vector A and B, respectively. The cosine similarity will range between 0 and 1 in the information retrieval problems.

Top-k similar documents D_{rel} to a query Q. Given a list of documents D, D_{rel} is defined as:

$$D_{rel} = argmax_k(argsort_{d \in D} \quad cosine(V[q], V[d])) \tag{3.2}$$

where $V[d]$ is a document-vector, and $V[q]$ is a query-vector. Thus, D_{rel} contains the top-k documents whose document-vector has the highest cosine similarity with the query-vector.

3.1.2 Query: expression and retrieval

Boolean Keyword Query. An index term is a word or expression, which may be stemmed, describing or characterizing a document. Let T be the set of all such index terms:

$$T = \{t_1, t_2, \ \ldots, \ t_m\}$$

A document is any subset of T. Let D be the set of all documents:

$$D = \{D_1, D_2, \ \ldots, \ D_n\}$$

A query is a Boolean expression Q in normal form:

$$Q = (W_1 \ \vee \ W_2 \ \vee \ \cdots) \wedge \ \cdots \ \wedge \ (W_i \ \vee \ W_{i+1} \ \vee \ \cdots)$$

where W_i is true for D_j when $t_i \in D_j$.

Query-as-Example is a set of documents where each document is represented by a bag of words d_1:{W_1, W_2, ..., W_n }. A word is transformed into a vector using a pre-trained embedding model (e.g. word2vec) to build in our algorithms. The query expressed as an example is represented by an example vector computed using a vectorization process (see above).

Retrieval (i.e., Boolean keyword query evaluation) seeks to find the set of documents that satisfy Q in two steps:

1. For each W_j in Q, find the set S_j of documents that satisfy W_j:
$$S_j = \{D_i \mid W_j\}$$
2. Then the set of documents that satisfy Q is given by:
$$(S_1 \cup S_2 \cup \cdots) \cap \cdots \cap (S_i \cup S_{i+1} \cup \cdots)$$

The implementation of retrieval varies according to the representation of the documents (query) content. In this work, a document (query) (i.e., term) is represented by a vector that aggregates the vectors of all its terms using a vectorization process (see above). As such, a document satisfies a query if their Cosine similarity (see above) is greater than a pre-defined threshold.

3.1.3 Query rewriting

Query Rewriting is an automatic transformation that takes a set of queries, data and other metadata, and yields a set of different queries, which produce the same results but execute with better performance (e.g., better precision). Chapter 2 introduced several query rewriting techniques used for exploring different kinds of data collections. The algorithms in this chapter propose query expansion (morphing and queries-as-answers) and query-by-example strategies for textual documents.

Query expansion. A Boolean keyword query can be expanded using terms synonyms, similar concepts from a given knowledge domain or from a semantic representation (ontology). This leads to three types of expanded queries:

– *Synonym based query*: extends each term of the initial query with a synonyms' set using a lexical thesaurus or a dictionary, for example WordNet[4]. The drawback is that using all possible synonyms can lead to have to include many noun synsets and collections for every term, that might not even represent exactly the user intention. Synonym based query includes synonyms which frequently occur in the top-k initial results. Accordingly, synsets of the initial query term, that do are not contained in the frequency matrix are filtered. For example, for a data collection regarding crisis management, assume that the frequency matrix identifies the terms "need", "ask", "take", "want"

[4]WordNet is a well-known resource for leveraging word synsets[5] to extend the original query, and many of the research works use WordNet [84, 85].

and "require". Consider the query "Need help". Let us focus on the term "need". Wordnet produces 16 synonyms:

```
{ "need", "require", "take",  "pauperlization",
  "pauperlism", "necessitate", "involve",
  "motivation", "demand", "penury", "postulate",
  "call for", "want", "indigence", "motive", "ask"}
```

They are filtered according to the frequency matrix content leading to 4 synonyms "ask", "take", "want", "require" that are then used for rewriting the query term "need".

– *Knowledge-based query*: expands the terms of an initial query with a local semantic word representation [86], assuming that local embeddings better capture the nuances of a topic-specific language [87].

  Producing a knowledge-based query relies on a frequency matrix of a documents collection, locally embedded keyword query terms and a locally embedded documents search space. For each query term vector, find top-m semantically related terms (i.e. their vectors are close in the search space). Look for similar terms which frequently occur in the top-k initial search space and filter those terms that are not in the frequency matrix.

– *Result set post-processing based query.* For each query term, retrieve the documents where the term occurs using the inverted index. Extract trigrams[6] from these documents' set. Select candidate trigrams (i.e., those sharing at least one term with the query terms and that are frequent) and use them to rewrite the initial query. Figure 3.2 shows an example of finding frequent trigrams of query term "help". The top frequent trigrams in the disaster-related corpus for the term "help" are extracted using the inverted index and retrieved results. The original query is rewritten by selecting trigrams from the candidate trigrams set.

Query Generation denotes the process of generating Boolean keyword queries that can potentially retrieve relevant documents to answer a query Q. The assumption is that it is possible to learn a binary decision tree [88] to generate Boolean queries that can morph Q using training data and

[6] *A continuous sequence of n terms from a given text sample is an n-gram. Trigrams are consecutive sequences of 3 words* (https://en.wikipedia.org/wiki/N-gram - Accessed the 10th July 2021)

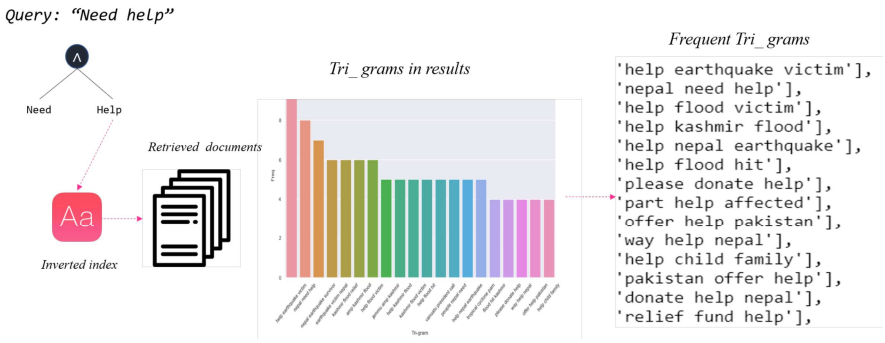

Figure 3.2: Trigrams based query rewriting example.

candidate terms. Decision tree learning is a method used in data mining[7]. To build the binary tree the idea is given items characterised by features (i.e., attributes), the first question is to choose the attribute that will best help to determine whether an item belongs to class a or b. Thus, in a binary decision tree each internal (non-leaf) node is labeled with an input feature. The arcs coming from a node labeled with an input feature are labeled with each of the possible values of the target feature. Each leaf of the tree is labeled with a class signifying that the data set has been classified by the tree into either a specific class. Intuitively, this process greedily builds binary trees using different attributes to choose the one that best classifies the items. Each branch in the tree can be translated into rules that combine feature values that can lead to items of class a or b.

In the case of Boolean query generation, two elements are used as reference: (i) the set of documents R_Q that represent the documents that are closest to Q, and (ii) the frequent terms of the top-k documents in R_Q. The objective is to find the combination of frequent terms that can best classify the documents in R_Q into relevant and irrelevant.

After this step, we have a binary decision tree that classifies documents into relevant and irrelevant, where each path from root to a leaf shows a classification rule. The paths that lead to the relevant documents class are used to state Boolean queries. These queries morph the initial Q since they combine terms of its relevant documents. For instance in the example shown in Figure 3.3 there is a binary decision tree that proposed four rules. For generating queries that can produce relevant documents, only the ones that lead to relevant documents are considered. Thus, two Boolean queries are defined:

[7] https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html

– Q1: Building ∧ Damage

– Q2: $\neg Building \wedge Historic$

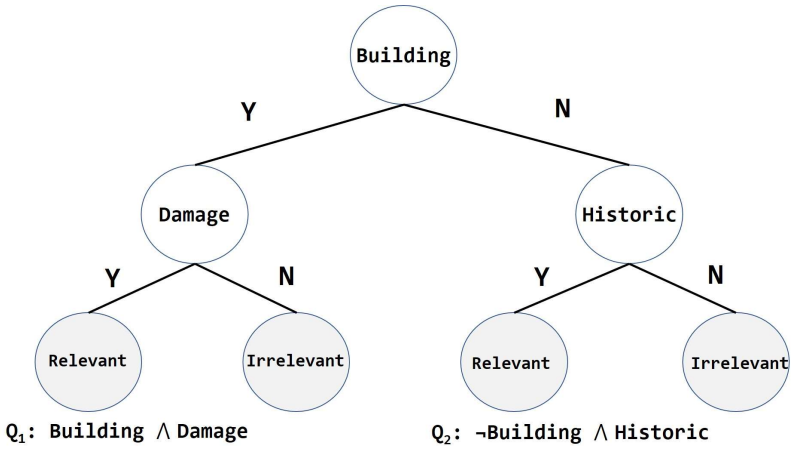

Figure 3.3: Example of a binary decision tree used for generating Boolean queries.

Ranking. *Given a query q and a collection D of documents that match the query, the problem is to rank, that is, sort, the documents in D according to some criterion so that the "best" results appear early in the result list displayed to the user*[8]. In our work ranking is performed on documents matching a query and on rewritten queries that can match relevant documents.

3.2 Query morphing

Query morphing is the process of rewriting conjunctive and disjunctive keyword queries, by adding terms, to increase the possibility of exploring the most number of items in a collection.

The proposed algorithm assumes that the data collection is as a set of documents modeled as bags of terms. Each document is represented as a point (vector) within a search space (see Figure 3.4). The initial Boolean keyword query (Q) is also a document represented as a vector that can be located within the search space. The result of Q corresponds to an area (R_q) within the search space where the distance between Q and the documents is lower than a certain threshold (see the rose area in the rectangle in the left side of Figure 3.4). Given Q, query morphing rewrites the query using

[8] https://en.wikipedia.org/wiki/Ranking_(information_retrieval)

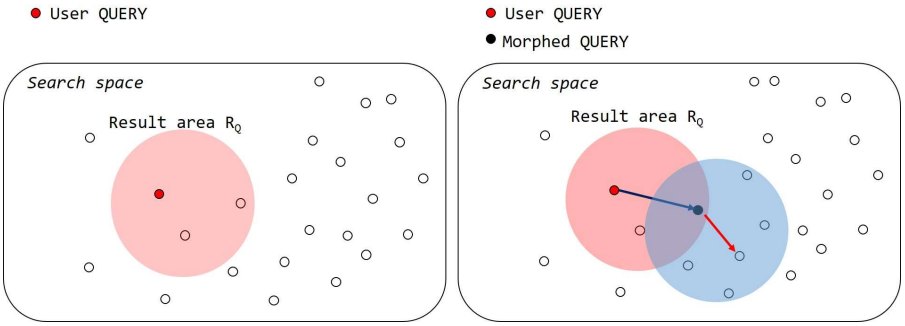

Figure 3.4: Query morphing general principle

"similar" terms appearing in the documents outside R_q, that is terms that are close to the terms of the documents in R_q (see the blue area in the rectangle in the right side of Figure 3.4). The objective to increase the possibility of exploring the most number of items in the search space.

Figure 3.5 shows the process of our query morphing algorithm consisting of three steps:

- Build the result area R_Q retrieving data samples (i.e., documents) that are similar to the original query, according to a given similarity function and a threshold.
- Generate a set of morphed queries using the terms contained in the data samples (i.e., documents) in the result area R_Q.
- Rank morphed queries, evaluate them and build result consisting of a set of tuples of the form $\{< MQ_1, DS_1 > ... MQ_i, DS_i >\}$, where MQ_i is a morphed query, and DS_i its associated result which is a data sample (documents sample).

Users can review the proposed queries with their samples and decide to start the loop again using as entry one of these queries until the generated morphed queries converge with the user expectations.

3.2.1 Query morphing algorithm

The pseudo-code that implements the proposed query morphing algorithm 1 is shown below. It relays in the word2vec model trained for computing a vector-based representation of textual content (documents and query) (see top-left(a) of Figure 3.6). Given a query Q modelled as a bag of terms, it starts computing a query vector using the pre-trained word2vec model (line 5) (see top-middle(b) of Figure 3.6).

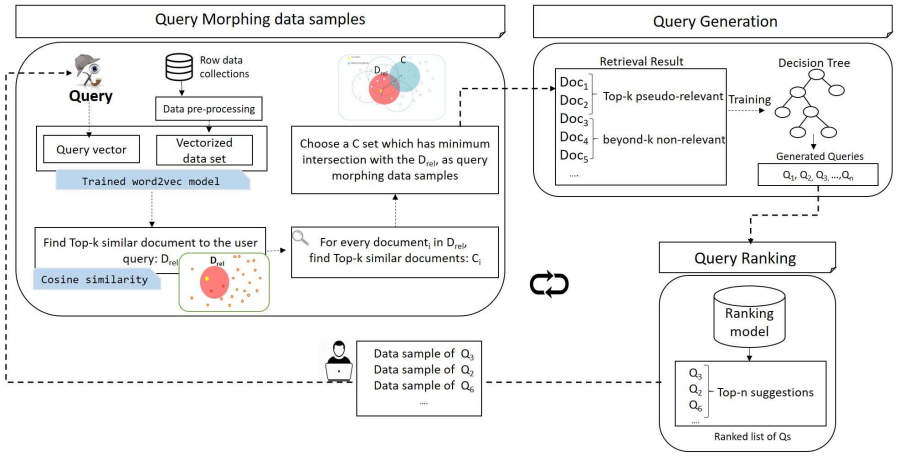

Figure 3.5: Query morphing.

The objective of the **stage A** of the algorithm is to explore the search space to build a result set R_Q retrieving the top-k similar documents to the initial query (lines 6 to 9) (see top-right(c) of Figure 3.6). The similarity between the query and the documents in the search space is calculated using the Cosine similarity.

Then the algorithm makes k iterations to generate morphed queries. In each iteration: (a) a document in the R_Q is transformed into a document vector dv_i; (b) the algorithm retrieves from the search space the top-k similar documents to dv_i ($R(dv_i)$), that is, those with the highest cosine similarity with dv_i (lines 10 to 16) (see bottom-right(d) of Figure 3.6).

At the end of the iteration, the algorithm produces k sets consisting of k documents. Then it computes the intersection of these sets, and it chooses a set with the minimum intersection with the R_Q (i.e., the main top-k) (see bottom-middle(e) of Figure 3.6), and it labels it as the top-k pseudo relevant documents set (see bottom-left(f) of Figure 3.6).

In **stage B** the algorithm applies the query generation function (see Algorithm 5) to compute the morphing queries set MQ_{set} consisting of Boolean queries that morph the initial one Q (lines 18 to 27).

Finally, the function query ranking is executed on the morphing queries set MQ_{set} (refer to (line 29) to produce a ranked list which is the final result of the query morphing algorithm (**stage C**).

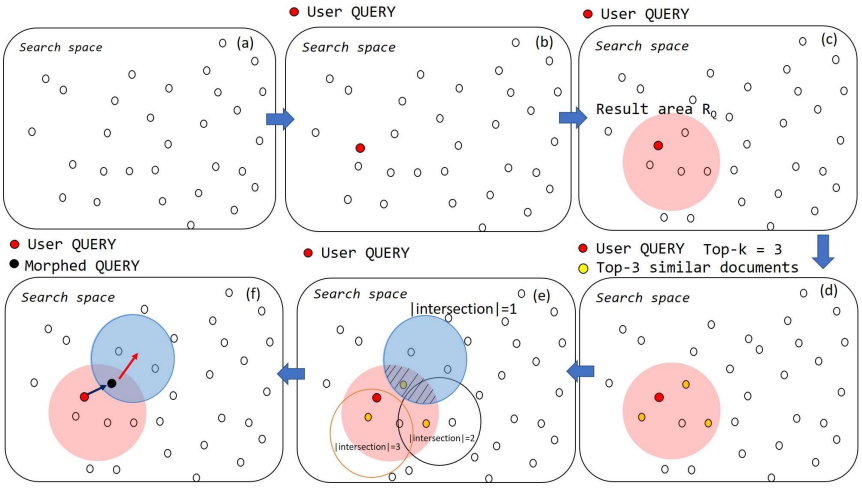

Figure 3.6: An example of query morphing process to find top-k pseudo relevant documents.

3.2.2 Example

In this section, we present an example of query morphing in a crisis scenario (such as earthquake, flood) introduced previously in Chapter 1.

For instance, consider Sophia, who works in humanitarian aid organizations and is responsible for extracting useful information for disaster relief. She is looking for missing people and/or people who lost their homes and need shelter. Sophia defines a simple query containing "missing, lost and shelter". The query denotes those documents containing the words: "missing", "lost" and "shelter". After reviewing initially retrieved documents, she wants to reformulate her initial query.

As shown in figure 3.7, when Sophia gives her query (Q: "missing AND lost AND shelter") to the query morphing algorithm and the query is morphed as follows:

```
Q1 = "missing AND lost AND help"
Q2 = "help AND people AND find"
Q3 = "shelter AND food"
```

As shown in figure 3.7, the targeted documents are contained in a results area, shown in the documents collection space. There are documents like those marked in green that should be interesting for the data consumer, but since they use related terms and not the correct terms expressed in the query, they are not put in the result area. The principle of query morphing extends the target area by including documents related to those in the initial result, but that is not included because they use "similar" but not the exact terms.

Algorithm 1: Query Morphing

Inputs: User query; Vectorized data set; Number of morphed query : K.
Outputs: Ranked list of generated morphed query: *top_k_morphed_queries*, data samples:S
    Initialize $Q = \{\}$ $S = \{\}$
STAGE-A;
    - query vector ← vector representation (User query)
    - **for** document in *dataset*:
        - $similarity_i$ ← cos($document_i$ vector, query vector)
    - *main_top_k* ← Choose top-K documents based on the similarity score
    - **for** document in *main_top_k*:
        - Consider $document_j$ vector as a potential query vector: *new_query*.
        - Calculate a similarity score between the *new_query* vector and all document of data set.
        - C_j ← Choose top-K documents based on the similarity score
STAGE-B: Query Generation;
    - *Top_k_pseudo_relevant* ← Choose a C_j which has minimum intersection with *main_top_k* as query morphing data samples.
    - *k_non_relevant* ← documents ranked beyond-k
    - Extract frequent terms that are semantically close to the user query from *Top_k_pseudo_relevant* as query morphing term candidates { $term_1$, $term_2$, ..., $term_M$ }: T.
    - K candidate term set ← K different sets of query term candidates s.t. { $candidate_1$, $candidate_2$, ..., $candidate_K$ } where $candidate_i$ is a set of query term candidates extracted from T.
    - Q ← Query Generation(*Top_k_pseudo_relevant*, *k_non_relevant*, K candidate term set) (Algorithm. 5)
STAGE-C: Query Ranking;
    - *top_k_morphed_queries*, S ← Query Ranking(Q)
    Return *top_k_morphed_queries*, S.

The query can be rewritten in our example, including the terms “help” or “find” consistently close to the terms shelter and missing.

Human-in-the-loop phase. Sophia can review formulated queries and their data samples to see which morphed query can produce the results that best respond to her expectations, and then she can leverage it for the next

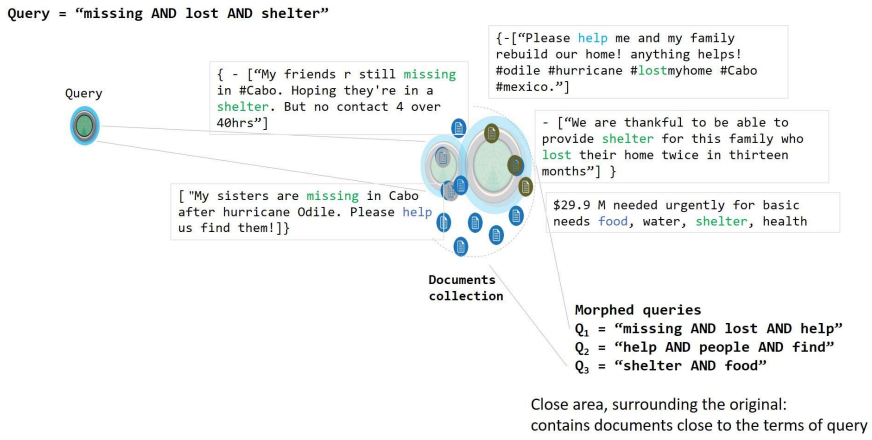

Figure 3.7: An example of query morphing in a crisis scenario.

iteration of her exploration.

3.3 Queries-as-answers

Given an initial Boolean keyword query, using the queries-as-answers strategy, the query is rewritten and transformed into several queries by extending it with general and more specific terms, synonyms, etc., and by exploiting the knowledge domain. The result is a set of possible alternative queries with associated sample results so that the user can choose which ones to execute.

As illustrated in figure 3.8 our queries-as-answers algorithm consists of three steps. Given a keyword query:

- retrieve top-k initial results using an information retrieval strategy;
- generate new queries using the top-k initial results and three query expansion approaches;
- rank generated queries, evaluate them, and return a result that associates queries with data samples.

Users can review the proposed queries with their data samples and start an exploration loop again until seeing satisfying queries and data samples.

3.3.1 Query-as-answers algorithm

The pseudo code of the algorithm 2 shows the process of our proposed query queries-as-answers. It consists of three stages.

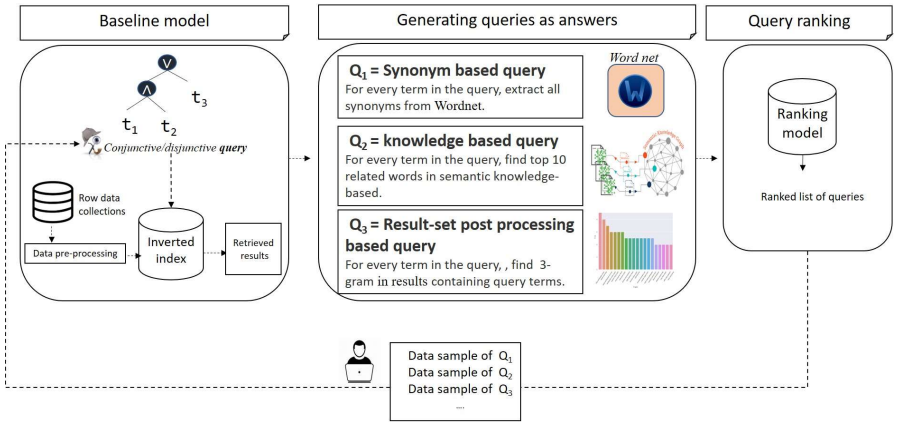

Figure 3.8: Queries-as-answer.

Stage A: Finding the top-k documents for an initial query Q. The algorithm retrieves the top-k relevant results using the Whoosh search engine, and then it extracts the top-m terms from the top-k initial result (lines 5 to 8).

Stage B: Expanding the initial query Q. In stage B, the algorithm uses three query expansion techniques to find proper terms to rewrite a query. The algorithm iteratively extracts synonyms from Wordnet for each term in the query using the frequency matrix to generate a synonym-based query. Those synonyms that do not occur frequently will be removed. Finally, candidate terms connect to the original query using OR and AND operations (refer to lines 12 to 22).

For each term in the query, the algorithm iteratively extracts similar terms from the word embedding model and then using a frequency matrix. Similar terms that are not frequent will be removed. Finally, candidate terms connect to the original query using OR and AND operations (refer to lines 24 to 36).

Result-set post-processing. For the result-set post-processing based query, the algorithm for each term in the query makes iterations on extracting documents which the term appear in them by using an inverted index and then extracting frequent trigrams from retrieved documents. Finally, candidate trigrams connect to the original query using OR and AND operations (refer to lines 38 to 46).

Stage-C: Ranking. Eventually, in stage C, generated queries will be sorted based on similarity score and precision using query ranking.

3.3.2 Example

Sophia issues an initial query containing "missing, lost and shelter", and she is willing to examine many retrieval results. She chooses queries-as-answers for exploring data collections. As shown in the figure 3.9, the result of the query-as-answers operation applied to a dataset of tweets can be used to produce three types of rewritten queries.

– The first one is synonym based query:

```
Q1 = "((missing OR lack OR absent))
       AND ((lost OR helpless OR confused OR suffer))
       AND ((shelter OR protection)))"
```

– The second one is a knowledge-based query:

```
Q2 = "(missing OR absent OR lack)
        AND (shelter OR protection)
        AND (victim)"
```

– The third one is result-set post-processing based query:

```
Q3 = "missing AND victim AND tourist hunkered shelter"
```

Figure 3.9: Frequent terms used in post-results.

Algorithm 2: Query-as-answers

Inputs: user query ; Inverted index; data set;
Outputs: Ranked list of query: *top_k_queries*
Initialize $Q = \{\}$ $S = \{\}$
STAGE-A;
- top-k initial result ← using Whoosh to retrieve top-k relevant results (user query, data set)
- F ← select top m terms from top-k initial result using tf-idf weights

STAGE-B;
Q_1**: Synonym based query**;
$Q_1 = \{\}$
- **for** each term in the query:
  - Extract all synonyms of $term_i$ from Wordnet $\{s_1, s_2, ...\}$: S
  - Consider only those synonyms which appear in the frequency term set as term candidates to add them to the query: $C = \{S \cap F\}$
  - Connect the $term_i$ with terms in C with the operator OR : $extended\_term_i$.
  - Connect the $extended\_term_i$ with Q_1 with the operator AND.
  - Append the Q_1 to Q.

Q_2**: Knowledge based query**;
$Q_2 = \{\}$
- **for** each term in the query:
  - Extract top-k related terms of $term_i$ from semantic knowledge-based by using Crisis NLP word2vec model $\{r_1, r_2, ..., r_k\}$: R
  - Consider only those related terms which appear in the frequency term set as term candidates to add them to the query: $C = \{R \cap F\}$
  - Connect the $term_i$ with terms in C with the operator OR : $extended\_term_i$.
  - Connect the $extended\_term_i$ with Q_2 with the operator AND.
  - Append the Q_2 to Q.

Q_3**: Result-set post processing based query**;
$Q_3 = \{\}$
- **for** each term in the query:
  - Extract documents which $term_i$ appear in them by using Inverted index : D.
  - Extract 3-grams from D which $term_i$ appear in them and connect 3-grams with the operator AND: $extended\_term_i$.
  - Connect the $extended\_term_i$ with Q_3 with the operator AND.
  - Append the Q_3 to Q.

STAGE-C: Query ranking;
- Sort the queries in Q based on query ranking metrics (similarity score and precision)

Return Q.

Human-in-the-loop phase. Sophia now can review a set of possible alternative queries with associated sample results so that she can choose which ones to execute. In addition, she can rewrite and expand her query. For instance, by using the terms "lack" or "absent", synonyms of the term "missing" or using the term "victim" that appears always close the term "lost". Since the recall of a query can be improved by expanding it with synonyms of the keywords of the query, Sophia can examine many results that are related to her initial query. In addition, she can reformulate her query by using frequency unigrams, bigrams, trigrams provided by the result-set post-processing query (see figures 3.10, 3.11 and 3.12).

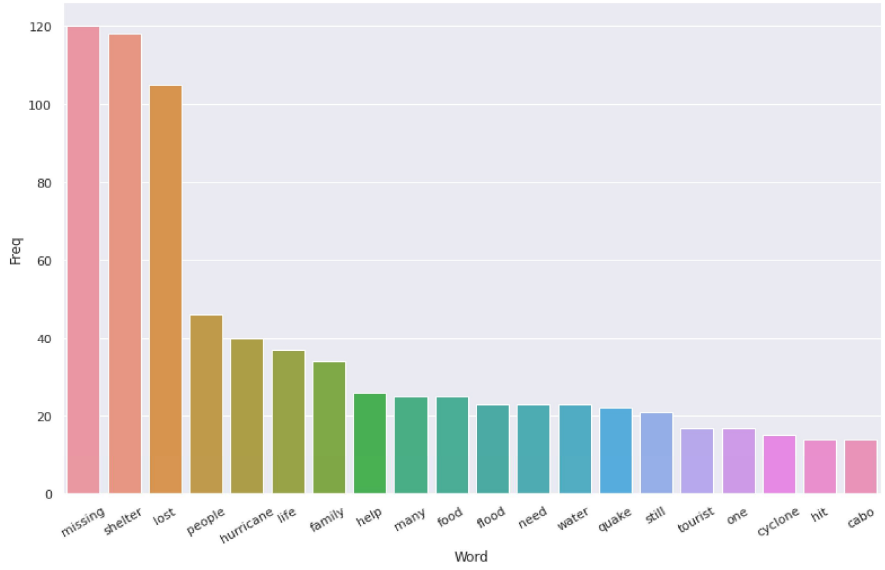

Figure 3.10: Frequency unigrams in results set.

3.4 Query-by-example

This algorithm exploits inherent characteristics of the data to infer the results that may not be easily expressed. Therefore, users give examples of the documents they want to find rather than writing queries. Given a set of documents representing an example, the algorithm looks for similar documents to build a result.

Figure 3.13 shows the process of our query-by-example algorithm consisting of three steps:

– Compute a vector using provided examples named query vector to

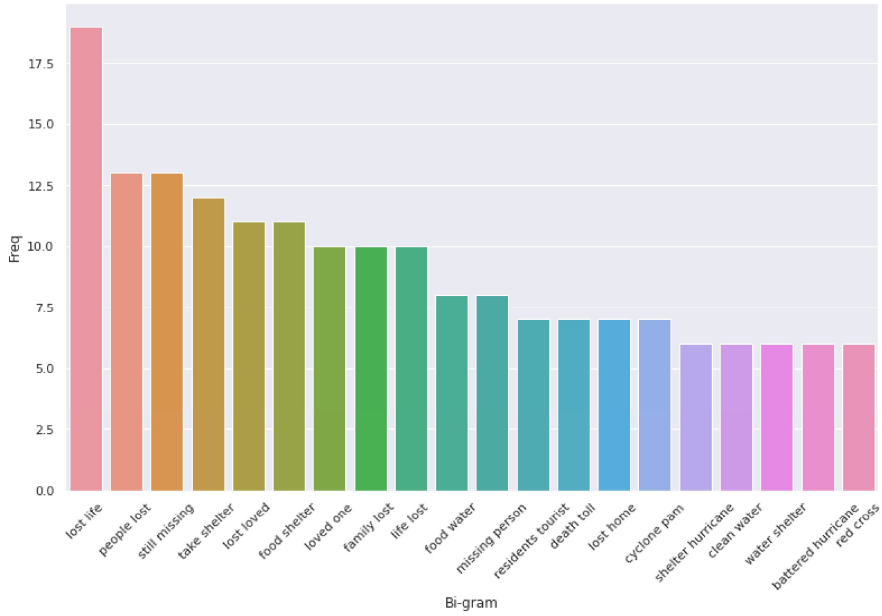

Figure 3.11: Frequency bigrams in results set.

retrieve related samples (i.e., documents).

- Generate a set of queries using retrieved data samples.
- Rank generated queries and build a result (output) consisting of queries with their associated data samples.

3.4.1 Query-by-example algorithm

The pseudo-code of the proposed query-by-example algorithm 3 consists of three stages.

Stage-A: Processing the example. An example consists of documents set that are given as a query. The objective is to compute example vectors (one per document in the query) using the pre-trained word2vec model (line 5). Compute a query vector summing the example vectors and computing their average. Given the query and the search space consisting of document vectors retrieve those that are close (i.e. similar) to the query vector. Then, select the top-k pseudo relevant documents which are top-k similar documents in the result set RQ (refer to lines 7 to 13).

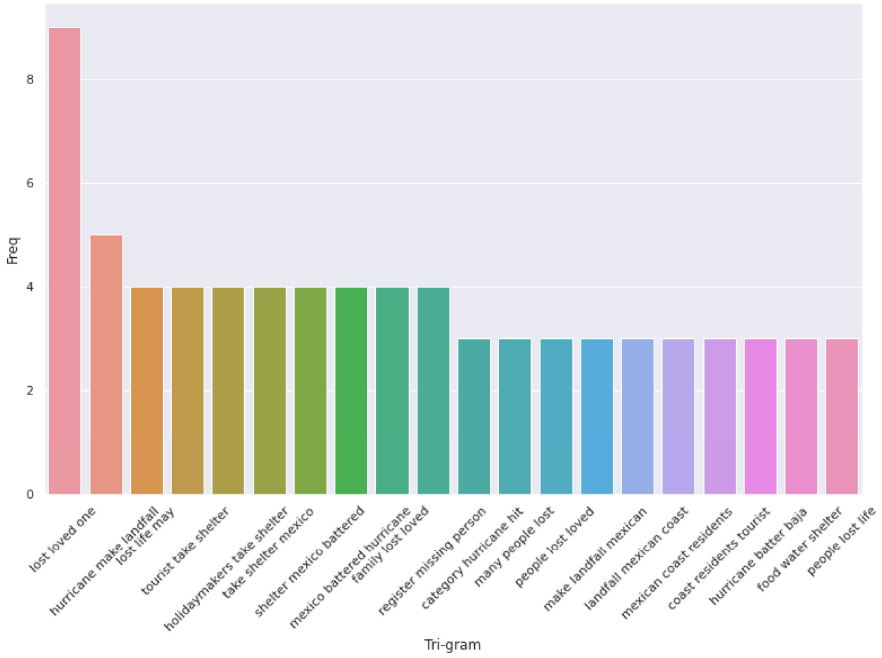

Figure 3.12: Frequency trigrams in results set.

Stage-B: Query Generation. Then algorithm calls query generation function (Algorithm 5). This function accepts three positive labels as input, negative labels and query term candidates and returns several queries as output. Top-k pseudo relevant documents are given as positive labels, documents ranked beyond-k as negative labels and frequent terms in the top-k pseudo relevant documents and given examples as query term candidates (lines 14 to 22). As an output, we have several Boolean queries.

Stage-C: Query Ranking. Finally, the query ranking function is executed (refer to line 24), and a ranked list of generated queries will be returned.

3.4.2 Example

Now Sophia is addressing a more delicate search, looking for injured or dead people. Sophia is aware of documents that should be present in her query output, so she would prefer to use query-by-examples exploration techniques which allows to express her request by giving examples of what she wants to find. For instance, as shown in figure 3.14, Sophia gives a set of examples consisting in the set of micro-documents:

Algorithm 3: Query-by-example

Inputs: User samples ; Vectorized data set; Number of query : K.
Outputs: Ranked list of generated query: *top_k_queries*; data samples: S
    Initialize $Q = \{\}$ $S = \{\}$
STAGE-A;
    - Convert all user data samples into vector representation using trained word2vec model : example-vector
    - Sum all example-vector followed by averaging them to obtain a query vector : *query_vector*.
    - Using Cosine similarity to calculate a similarity score between the *query_vector* and all the elements of data set.
STAGE-B: Query Generation;
    - *Top_k_pseudo_relevant* ← Choose top-K documents based on the similarity score.
    - *k_non_relevant* ← documents ranked beyond-k
    - T ← Extract frequent terms from *Top_k_pseudo_relevant* documents and from given examples as query term candidates { $term_1$, $term_2$, ..., $term_M$ }.
    - K candidate term set ← K different sets of query term candidates s.t. { $candidate_1$, $candidate_2$, ..., $candidate_K$ } where $candidate_i$ is a set of query term candidates extracted from T.
    - Q ← Query Generation(*Top_k_pseudo_relevant*, *k_non_relevant*, K candidate term set) (Algorithm. 5)
STAGE-C: Query Ranking;
    - *top_k_queries*, S ← Query Ranking(Q)
    Return *top_k_queries*, S.

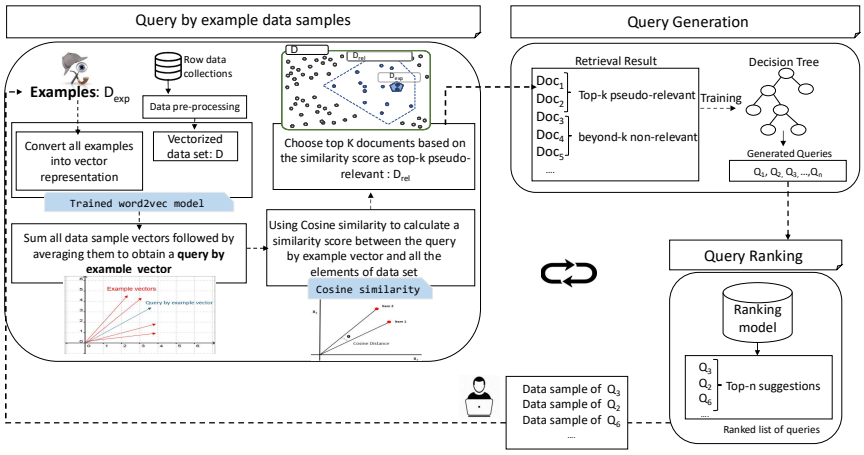

Figure 3.13: Query-by-example.

```
{"One dead several injured as bus
  overturns in Orangi Town",
 "Six people critically injured in California earthquake",
 "Many people killed and injured in powerful quake"}.
```

According to the samples provided, the user is looking for tweets about individuals who have died or been injured in a disaster. As shown on the right side of the figure 3.14, the result of the query-by-examples operation applied to a dataset of tweets, is the following set of queries:

```
Q1 = "dead AND people"
Q2 = "dead AND injured"
```

Along with their data samples (4 documents):

```
{"At least 40 dead in Pakistan blast",
 "Over 400 People Dead, 1000 Injured As
  Massive Earthquake Hits Nepal",
 "At least 328 people have been confirmed dead and
  more than 450 injured",
 "Massive #earthquake in #Pakistan
  - At least 283 #killed and 400 #injured"}
```

Human-in-the-loop phase. Sophia can continue her exploration by using return result as query examples to find new results containing further similar documents.

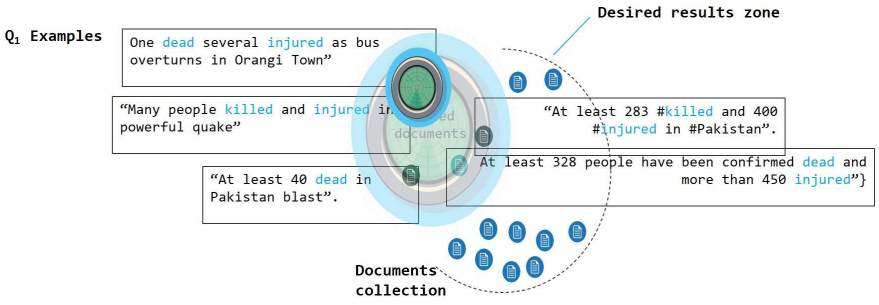

Figure 3.14: An example of query-by-example.

3.5 Experimentation

We conducted experiments on several crisis-related queries to evaluate our algorithms. This section describes the experiments, including the data collections that have been used and experimental setups. Experiments were performed on the Google Colaboratory cloud platform[9] (i.e. Google Colab) with 25GB NVIDIA GPU RAM, Xeon Processors @2.3Ghz.

3.5.1 Experimental setup

Microblogs related to crisis events. To evaluate the different exploration techniques, we used the CrisisNLP data set. The collection contains 35648 labelled data related to the crisis tweets posted during 8 disaster events (see table 3.1).

Table 3.1: Summary of data collections related to eight crisis events.

Crisis Name	Country	Number of Tweets	Year
Nepal Earthquake	Nepal	12489	2015
Typhoon Hagupit	Phillippines	9675	2014
Cyclone PAM	Vanuatu	2613	2014
Chile Earthquake	Chile	2453	2014
Hurricane Odile	Mexico	2196	2014
California Earthquake	USA	2196	2014
Pakistan Floods	Pakistan	2013	2014
Pakistan Earthquake	Pakistan	2013	2013
Total Number		35648	

After performing data pre-processing and removing duplicates and near-duplicates, a total of 10044 tweets remained (described in Section 4.1.1). As discussed earlier, the tweets are labelled into various informative classes,

[9] https://colab.research.google.com/

and we modified the labels of tweets such that each crisis event data set has two labels: *Event* (a situation produced during a disaster) and *Action* (represents reactions to events). Table 3.2 shows the number of tweets for each class. All of the experiments in this chapter were conducted using this data set.

Table 3.2: Description of the classes in the data collection

Class	Total label	Description
Event	3842	Tweets reporting occurrence and happening of events during the crisis. Reports deaths, injuries, missing, found, or displaced people,infrastructure and utilities damage
Action	6202	Tweets reporting responses and measures taken by people during crisis. Messages containing donations or volunteering offers also sympathy-emotional support
Total number	10044	

3.5.2 Synthetic queries

This section describes the process we defined for constructing synthetic queries with different sizes used to evaluate our proposed query morphing.

Similar to the dataset used for evaluation that includes two classes (i.e. Event and Action), synthetic queries consist of two classes. Queries created using Event class vocabulary are in event-query class, and queries constructed using Action class vocabulary are in action-query class. Table 5.9 illustrates 15 terms that occurred frequently in the Event and Action class, respectively. We consider them as seed terms to construct synthetic queries.

Table 3.3: Candidate terms of each class for building synthetic queries.

Class	Seed terms
Event	{ dead, injured, missed, trapped, displaced, kill, victim, damaged, building, road, bridge, village, destroy, communication, home }
Action	{ donation, shelter, food, water, money, medical, volunteer, clothing, pray, heart, hope, sad, tears, crying, thoughts }

According to study [89], users often pose two or three query terms on average. Therefore, we constructed queries with sizes one, two and three. Table 3.4 summarizes the number of synthetically generated queries with sizes one, two and three in each class. Mathematically, since the order of terms in a query does not matter, a combination equation can be used to calculate several synthetically generated queries as:

$$\binom{n}{k} = \frac{n(n-1)...(n-k+1)}{k(k-1)...1} \tag{3.3}$$

where n is number of seed terms in a class and k is the size of query. For instance, the number of queries with size three using seed terms of event class can be described as a 3-combination ($k = 3$) of terms from a 15 seed terms ($n = 15$). The 3 terms of the query are all distinct, and the order of terms in the query does not matter. There are $\binom{15}{3} = 455$ such combinations. As shown in table 3.4, we evaluate our proposed exploration technique using 1150 queries with different size.

Table 3.4: Number of synthetic queries for each class.

Class	# of query with one term	# of query with two terms	# of query with three terms	# of query in each class
Event-query	15	105	455	575
Action-query	15	105	455	575
Total query	30	210	910	**1150**

3.5.3 Evaluation metrics

We choose four performance metrics for the evaluation of our technique that are frequently used in the literature. [90–93].

Mean Precision@K (MP@K). Precision@K calculates the occurrence of relevant items in the top-k retrieved documents of a query. Mean Precision@K is the mean of Precision@K for a set of queries (Q). MP@K can be defined as follows:

$$P@K = \frac{|\{relevant\ documents\} \cap \{topK\ retrieved\ documents\}|}{|K|} \tag{3.4}$$

$$MP@K = \frac{\sum_{q \in Q} P@K(q)}{|Q|} \tag{3.5}$$

Here, K refers to several top results considered, and Q is the set of all queries.

Mean Recall@K (MR@K). Recall@K refers to the percentage of relevant items that are correctly retrieved for a query in the top-K results. Mean Recall@K is the mean of Recall@K for a set of queries (Q). It can be defined as follows:

$$R@K = \frac{|\{relevant\ documents\} \cap \{topK\ retrieved\ documents\}|}{|\{relevant\ documents\}|} \tag{3.6}$$

$$MR@K = \frac{\sum_{q \in Q} R@K(q)}{|Q|} \tag{3.7}$$

Here, K refers to number of top results considered, and Q is the set of all queries.

F-score@K combines Precision and Recall of the model and use it as a standard measure of performance. F-score is defined as the harmonic mean of Precision and Recall.

$$Fscore = \frac{2 \times precision \times recall}{(precision + recall)} \tag{3.8}$$

To calculate F-score@K we use MP@K and MR@K instead of precision and recall, respectively. An F-score can have a maximum value of 1.0, which indicates perfect precision and recall, and a minimum value of 0.

Mean Similarity@K' (MS@K'). Similarity@K' calculates the similarity between top K' generated queries and initial query by using Cosine similarity.

$$S@K' = \frac{\sum_{q \in Q} Cosine\ similarity(q, Q')}{|K'|} \tag{3.9}$$

Here K' refers to the number of top generated queries, Q is the set of top generated queries considered, and Q' denotes the initial query.

3.5.4 Query morphing experimental results

Experimental Results. Table 3.5 presents the performance details of our query morphing technique for synthetic queries with sizes one, two and three. For each synthetic query, we considered top 5 queries generated by query morphing (5750 morphed queries generated for the 1150 synthetic queries) to calculate MS@5, and for each generated query, we retrieved top 50 related data samples to calculate MP@50, MR@50 and F-score@50. It is expected that when an event-query is given, event-tweets are extracted, and vice versa for the action query. Hence the measure of relevancy in our experiments is that query and obtained results belong to the same class.

As evidence from Table 3.5, generated action-queries performed considerably better than event-queries in terms of mean Precision@50 and Recall@50.

Two reasons can explain the poor performance of generated event queries. First, the vocabulary gap among the seed terms used for constructing synthetic queries and event-tweets. Some of the use of event seed terms in both event tweets and action tweets. This problem shows the importance of selecting proper terms to generate the initial query. Second, probably, the crisis data collections contain much fewer event-tweets, pre-trained embedding (i.e. CrisisNLP word2vec model) cannot capture event-tweets context very well.

In Table 3.5, we observe that adding terms in synthetic action-queries increases the performance of generated morphed queries in terms of all measures. In contrast, adding terms in event queries decreases the performance of generated morphed queries. This proves that the seed terms of event class are not as high quality as seed terms of action class.

Table 3.5: Query morphing performance based on mean precision@50(MP@50), mean recall@50(MR@50), F-score@50 and mean similarity@5(MS@5).

Class	# of term in synthetic queries	# of synthetic queries	# of morphed queries	MP@50	MR@50	F-score@50	MS@5
Event	1	15	75	0.53	0.0068	0.013	0.56
	2	105	525	0.50	0.0065	0.012	0.54
	3	455	2275	0.44	0.0057	0.011	0.54
Action	1	15	75	0.84	0.0067	0.013	0.56
	2	105	525	0.86	0.0069	0.013	0.52
	3	455	2275	**0.92**	**0.0074**	**0.014**	**0.59**
Total		1150	5750				

Qualitative analysis. We now provide a qualitative analysis of our query morphing techniques via 4 examples. Table 3.6 illustrates the top 5 generated morphed queries for four different queries with various length and their performance. We see that the performance of morphed queries in terms of precision@50 is above 0.90 except in the second example. In the second example, for the event-query ("displaced and trapped"), the system used terms such as "thousands" and "million" that often occur with term query as candidate terms. However, "thousands" and "million" are general terms that can be used in both event tweets and action tweets. Hence, using these terms cannot only help find event tweets but also cause decrease precision. Also, Table 3.6 shows that the similarity score on average is about 0.50 (except example 1). One of the reasons is that, in query morphing, we want to let the user explore a broader data spectrum and see new terms surrounding a closed region of the original query. Therefore, the terms used in the query morphing may not be completely similar to the initial query.

Table 3.6: Examples of the top 5 generated morphed queries for four different queries and their performance.

N	Synthetic query	Query class	Morphed queries	Precision@50	Similarity
1	(injured and victim)	Event	(injured AND people)	0.90	0.91
			(injured AND hospital)	0.96	0.86
			(injured AND family)	0.86	0.879
			(injured AND say)	0.94	0.90
			(injured AND please)	0.84	0.87
2	(displaced and trapped)	Event	(thousands AND displaced)	0.68	0.42
			(thousands AND least AND killed)	0.94	0.46
			(people AND million)	0.62	0.59
			(people AND going)	0.22	0.60
			(affected AND people)	0.02	0.43
3	(donation)	Action	(nepal AND help AND donation)	0.98	0.47
			(nepal AND help AND donating)	0.98	0.47
			(help AND appeal)	0.96	0.49
			(nepal AND help AND appeal)	0.98	0.51
			(nepal AND help AND fund)	1.0	0.50
4	(pray and donation and shelter)	Action	(help AND need AND shelter)	0.80	0.48
			(help AND need AND donation)	0.96	0.53
			(help AND safe)	0.96	0.48
			(help AND need AND urgently)	0.98	0.46
			(help AND affected)	0.98	0.54

3.5.5 Query-by-example experimental results

The experimental setup for evaluating the queries-by-example algorithm, uses synthetic queries that include a set of data samples (i.e. examples) and the size of the synthetic query shows how many data samples are in that query. We evaluate our algorithm using synthetic queries with different sizes. To obtain multiple sets of examples, we consider 4 different sizes of synthetic queries (i.e. {1,3,5,10}) and then we generate 50 different synthetic queries in each size. In other words, we construct 50 synthetic queries with size 1, 50 queries with size 3, 50 queries with size 5 and 50 queries with size 10 for each class(400 synthetic queries for both classes). To prepare synthetic event-query, we randomly select a set of tweets ({1,3,5,10}) from the event-tweets data set and similarly, for the action-query, we do this process.

Experimental Results. Table 3.7 presents the performance details of our query-by-example technique for synthetic queries with size {1,3,5,10}. For each synthetic query. we retrieved the top 50 related data samples to calculate MP@50, MR@50 and F-score@50 and MS@50.

As shown in Table 3.7, action-queries generated using given examples performed considerably better than event-queries in terms of mean Precision@50, which indicates that event-tweets are much more difficult to retrieve. The reason is that the action-tweets class is more extensive (almost two times) than the event-tweets class. Moreover, we observe that adding

examples in both synthetic action-queries and event-queries increases the performance of generated morphed queries regarding mean Precision@50, mean Recall@50 and F-score@50. This result is expected when the user provides more examples, because the algorithm will better determine what the user is looking for. In contrast, adding examples in both classes decreases the mean similarity@50, although this decrease is not significant.

Table 3.7: Query-by-example performance based on mean precision@50(MP@50), mean recall@50(MR@50), F-score@50 and mean similarity@50(MS@50)

Class	# of given examples	# of synthetic query	MP@50	MR@50	F-score@50	MS@50
Event	1	50	0.71	0.0092	0.018	0.84
	3	50	0.74	0.0096	0.019	0.80
	5	50	0.77	0.0100	0.019	0.80
	10	50	0.75	0.0097	0.019	0.79
Action	1	50	0.94	0.0075	0.015	0.84
	3	50	0.97	0.0078	0.015	0.82
	5	50	0.98	0.0079	0.015	0.82
	10	50	0.98	0.0079	0.015	0.82
		400				

Qualitative analysis. We now provide a qualitative analysis of our query-by-examples techniques via 2 examples. Table 3.8 states the top 3 retrieved data samples for two different queries -with sizes one and three respectively- and their performance.

Table 3.8: Examples of query-by-example performance for two different queries with various lengths.

N	Examples as query	Retrieved data samples	P@50	S@50
1	['A sad day for Nepal. Let's sympathies with the families who lost love ones and pray for them']	['With love and thoughts for the lives lost and the ones still struggling', 'We pray for the beautiful people; nation of nepal. For those injured ', 'my thoughts and prayers go out to those who lost their lives']	0.90	0.88
2	['a lot of villages are almost impossible to get to'] ['Nepal left severely damaged with more than 100 dead '] [' Over 110 volunteers tracing thousands of kms of mountain roads and trails for Nepal right now on #Nepa']	['Roads are so damaged in Nepal it took a CNN crew two hours to go 20 miles','Villages Near Nepal Earthquakes Epicenter are Desperate','One of the main concerns now is what will happen with the several thousand buildings that were already damaged by the last earthquake']	0.68	0.80

3.5.6 Queries-as-answers experimental results

Experimental results. Table 3.7 presents the performance details of our queries-as-answers algorithm for synthetic queries with sizes one, two and

three. For each synthetic query, one synonym based query, one Knowledge-based query and one result-set processing query generated (3450 queries generated for the 1150 synthetic queries). For each generated query, we retrieved the top 50 related data samples to calculate MP@50, MR@50 and F-score@50.

As evidence from Table 3.9, generated action-queries achieved a better mean Precision@50 score. However, event-queries performed better than action-queries in terms of mean recall@50 and F-score@50. Also, we see that adding terms in synthetic queries increases the performance of generated queries in terms of mean Precision@50 (best MP@50 of 0.93 for action-query with three terms and best MP@50 of 0.72 for event-query with three terms), mean recall@50 (best MR@50 of 0.0093 for event-query with three terms and best MR@50 of 0.0074 for action-query with three terms) and F-score@50 (0.018 for event-query and 0.014 for action-query). On the other hand, we observe that one-term synthetic queries achieved the best similarity score (0.91 for action-query and 0.89 for event-query).

Table 3.9: Queries-as-answers performance based on mean precision@50(MP@50), mean recall@50(MR@50), F-score@50 and similarity.

Class	# of term in queries	# of synthetic queries	# of generated queries	queries as answer type	MP@50	MR@50	F-score@50	Similarity
Event	1	15	15	Synonym based query	0.53	0.0068	0.013	0.74
			15	Knowledge based query	0.58	0.0075	0.014	0.89
			15	Result-set processing query	0.65	0.0084	0.016	0.73
Event	2	105	105	Synonym based query	0.50	0.0065	0.012	0.54
			105	Knowledge based query	0.67	0.0087	0.017	0.62
			105	Result-set processing query	0.69	0.0089	0.017	0.57
Event	3	455	455	Synonym based query	0.47	0.0061	0.012	0.62
			455	Knowledge based query	0.72	**0.0093**	0.018	0.71
			445	Result-set processing query	0.71	0.0092	0.018	0.66
Action	1	15	15	Synonym based query	0.84	0.0067	0.013	0.77
			15	Knowledge based query	0.85	0.0068	0.013	**0.91**
			15	Result-set processing query	0.69	0.0055	0.011	0.76
Action	2	105	105	Synonym based query	0.86	0.0069	0.013	0.52
			105	Knowledge based query	0.90	0.0072	0.014	0.55
			105	Result-set processing query	0.79	0.0063	0.012	0.53
Action	3	455	455	Synonym based query	0.87	0.0070	0.013	0.60
			455	Knowledge based query	**0.93**	0.0074	0.014	0.64
			455	Result-set processing query	0.82	0.0066	0.013	0.61
Total		1150	3450					

Figure 3.15 depicts a comparison between synonym based query, knowledge-based query and results set post-processing query performance. We observe that knowledge-based queries outperformed other queries, achieving an MP@50 score of 0.77 and a similarity score of 0.72.

Qualitative analysis. We now provide a qualitative analysis of our queries-as-answer techniques via 4 examples. Table 3.10 illustrates the synonym based query, knowledge-based query, and results set post-processing query generated for four different queries with various length and their performance.

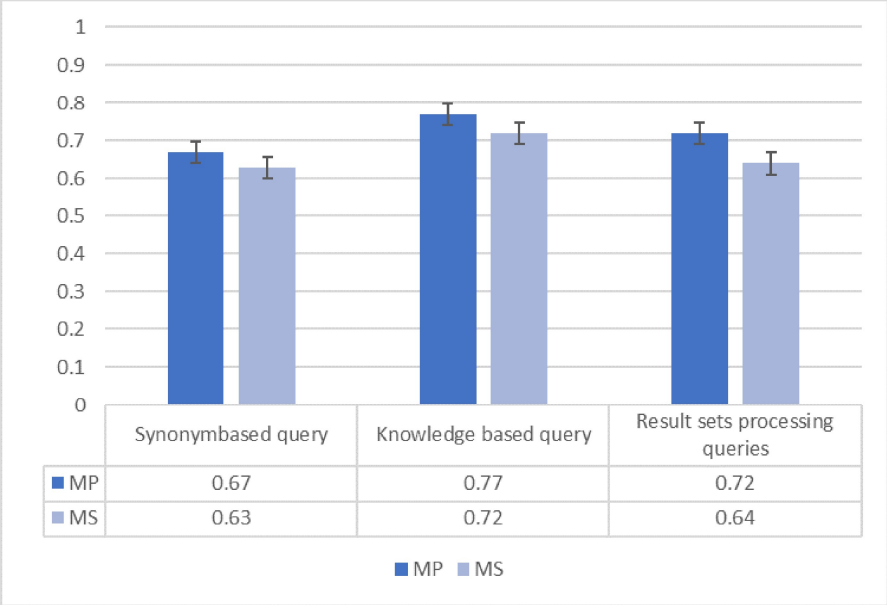

Figure 3.15: A comparison among synonym based query, knowledge based query and results post-processing query .

We notice that the performance of generated queries in terms of precision@50 is more significant than 0.80. Also, we can observe that the average similarity score is more than 0.85. For the first example, the ("injured and victim") system could not generate any synonym based query. The reason is that synonyms of terms injured and the victim did not frequently occur in the dataset because this system ignored them and did not consider them as candidate terms for expanding query.

3.5.6.1 Comparison and discussion

This chapter described and experimented with query morphing, queries-as-answers and query-by-example exploration algorithms for identifying event-tweets and action-tweets. We now discuss and compare the relative performance of these three types of techniques.

Figure 3.16 shows the performance of different techniques in both event and action classes. Mean Precision(MP) and Mean Similarity(MS) represent average of precision and average of similarity taken from table 3.5, 3.9 and 3.7. We have made the following observations.

As seen in figure 3.16, query-by-examples achieved the highest overall mean precision and similarity among all methods (e.g. 0.85 for mean precision,

Table 3.10: Examples of query-by-example performance for four different queries with various lengths.

N	Synthetic query	Query label	Queries as answer	Precision@50	Similarity
1	(injured and victim)	Event	[-]	–	–
			[(injured OR killed) AND (victim)]	0.94	0.96
			[(victim AND (dozens AND injured AND critically)), (victim AND (buildings AND destroyed AND injured)), (injured AND (prayer AND victim AND family))]	0.93	0.87
2	(displaced and trapped)	Event	[(displaced OR move OR can OR fire OR displace) AND (trapped OR pin OR trap)]	0.80	0.82
			[(displaced OR population OR shelter OR confirmed OR dead) AND (trapped OR rescued)]	0.80	0.83
			[(trapped AND (dead AND displaced AND cyclone), (trapped AND (injured AND nearly AND displaced)), (displaced AND (travel AND trapped AND tourist)]	0.86	0.82
3	(donation)	Action	(donation OR contribution)	0.98	0.88
			(donation OR appeal OR donate)	1.0	0.85
			[(urgent AND appeal AND donation), (appeal AND donation AND victim), (donation AND received AND government)]	0.92	0.77
4	(pray and donation and shelter)	Action	[(pray OR beg) AND (donation OR contribution) AND (shelter OR protection)]	0.94	0.90
			[(pray OR prayer OR people OR god OR right OR everyone OR family OR hope) AND (donation OR appeal OR donate) AND (shelter OR need)]	0.98	0.86
			[(donation AND shelter AND (please AND pray AND people), (donation AND shelter AND (pray AND flood AND victim),(donation AND pray AND (take AND shelter AND mexico)]	0.85	0.93

0.81 for mean similarity). On the contrary, query morphing does not perform well either in terms of precision or similarity (e.g. 0.68 for mean precision, 0.55 for mean similarity). We believe that a naive user who is not familiar with the dataset, and consequently cannot provide good examples to run a query-by-example technique, can utilize query morphing and queries-as-answer as an operation which supports the user by assisting the navigation in the data space to find interesting queries and data samples. After that, she can choose the interesting items among retrieved data as good examples and give them to the query-by-example algorithm to find more data objects like them. HILDEX provides such framework (see Chapter 3).

Figure 3.17 and 3.18 also illustrate performance of different techniques in both event and action classes separately. We observe that all exploration techniques perform better in the action-tweets data set rather than event-tweets. One of the reasons probably is action-tweets class is larger (almost two times) than the event-tweets class.

3.6 Conclusions

This chapter introduced exploration algorithms to rewrite queries that can fully explore textual data collections. They are complementary query rewriting techniques, where expanding a Boolean keyword query can help adjust the terms used for exploring a textual data collection, and combine terms

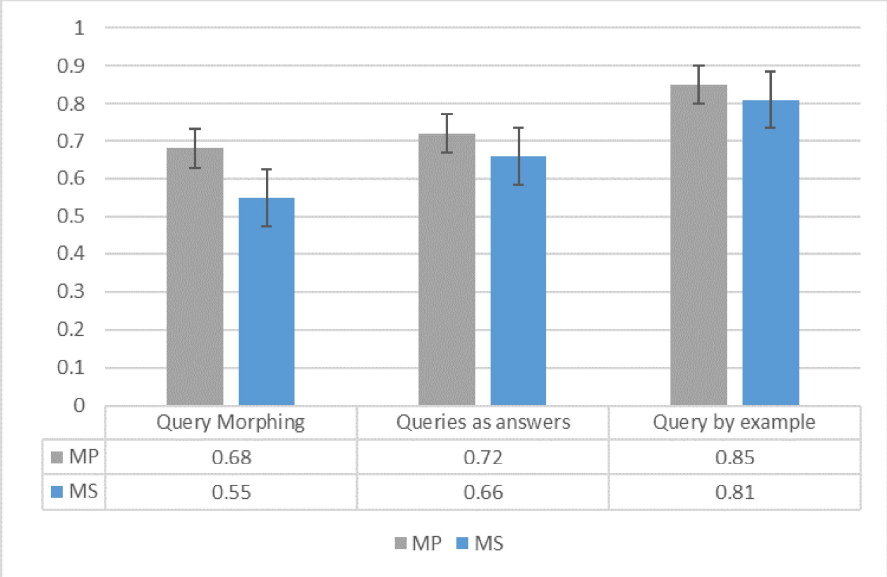

Figure 3.16: Mean Precision (MP) and Mean Similarity (MS) of all exploration techniques.

into new queries. Rewritten queries can match different sets of documents with associated quality scores.

Existing query morphing algorithms are based on the principle that an initial query matches (top-k) relevant documents. These documents are processed to extract terms that are then used to rewrite the initial query. Therefore the perimeter of the explored space is given by the top-k relevant results. The reason is that the objective is only to target relevant documents and not to expand the exploration space. Instead, the principle of our query morphing algorithm is to rewrite the initial query with terms that allow a user to explore a more expansive space (beyond the top-k relevant space).

Queries-as-answers algorithms rewrite an initial query and produce similar alternative queries that can better match relevant results. The objective is to find the one(s) that will perform better concerning some set of metrics. The queries-as-answers algorithm proposed in this work uses different techniques to generate queries. It associates queries to sample results so that a user can choose those that match documents that fulfill target expectations. The rewriting techniques also use semantically similar terms provided by general vocabularies and specific knowledge domain ones. Furthermore, these techniques combine unigrams, bigrams, and trigrams to increase the documents exploration space.

The query-by-example approaches have addressed the need to let users

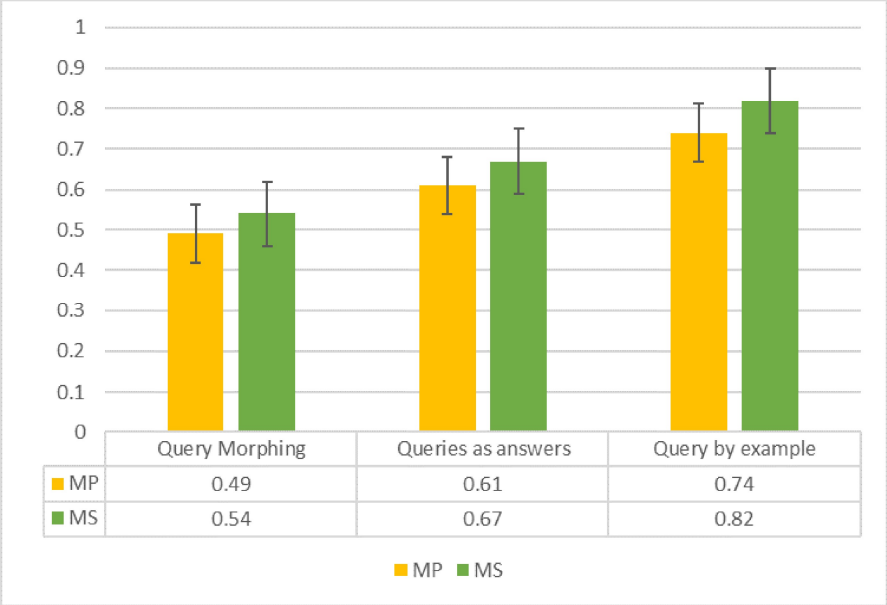

Figure 3.17: Comparison MP and MS based on event class.

express information requirements without writing queries. The idea is to retrieve items (i.e., documents) similar to an initial example. We rely on this principle to propose an algorithm to retrieve documents and produce queries that can target documents similar to the user example and those similar to it.

Finally, an important feature of the algorithms is that their process include the user who can choose the proposed queries according to her interests and the produced results that target her expectations.

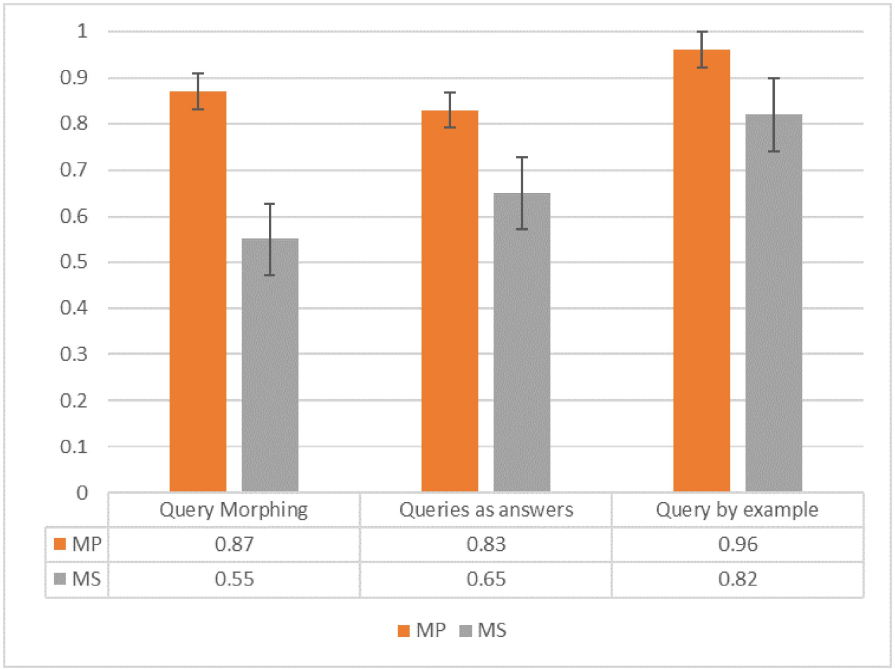

Figure 3.18: Comparison MP and MS based on action class.

CHAPTER 4

HILDEX: Human-In-the-Loop Data EXploration framework

This chapter introduces HILDEX, a human-in-the-loop data exploration framework for exploring textual data collections. It is based on a data exploration workflow with phases that successively execute query morphing, queries-as-answers and query-by-example algorithms (see Chapter 3) according to a user's feedback.

The chapter is organized as follows. Section 4.1 presents HILDEX's general architecture with the main components and their functions. Section 4.2 introduces the general exploration workflow implemented by HILDEX, describing its main phases and showing the way human intervention is weaved across the phases for guiding the exploration process until it converges. Section 4.3 shows, through an example, the way HILDEX can be used for exploring data collections in a target application related to crisis management challenges. Section 4.4 concludes the chapter by highlighting HILDEX's strong points and limitations.

4.1 HILDEX general architecture

HILDEX is a textual data exploration framework that combines query-morphing, queries-as-answers and query-by-example techniques for supporting human-guided data exploration workflows. Figure 4.1 shows HILDEX's general functional architecture consisting of components implementing:

– The general exploration workflow (Scheduler). It is the main component implementing the exploration workflow that schedules the exploration phases. Its execution is guided by feedback provided by a user (Feedback processor).

– Exploration algorithms (Query Morpher, Query as Answers Generator) and (Query as Example Generator).

– Document and query processing tasks (Document Content Processor and Query Evaluator). The Document Content Processor component uses classic information retrieval (terms frequency matrices, inverted indexes), natural language processing techniques (word embedding) and machine learning models (neural networks) to represent and process textual content. The Document Content Processor processes textual data collections and queries to support the exploration process.

– Query and results generation (Results Generator and Results Ranking). The user's decision-making regarding the need to explore or converge the process depends on a ranking component (Results Ranking), that orders the results and helps the user better observe whether results are those fulfilling her expectations.

The following sections further describe the functions of the main components of HILDEX.

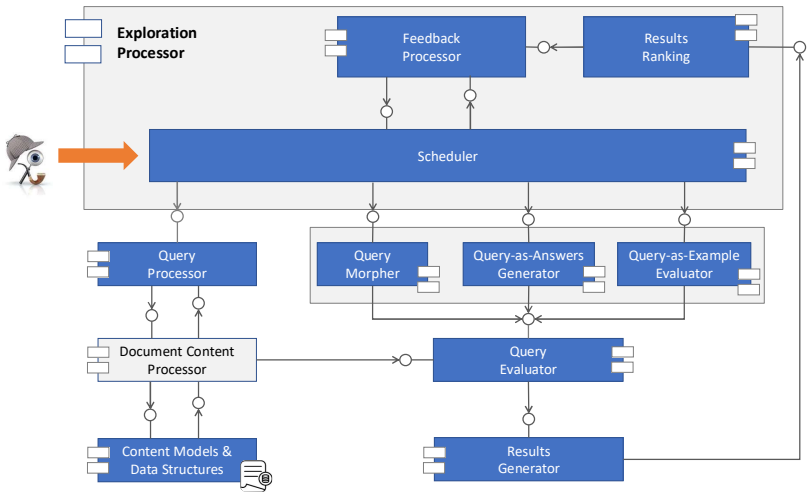

Figure 4.1: HILDEX general architecture (UML component diagram)

4.1.1 Document Content Processor

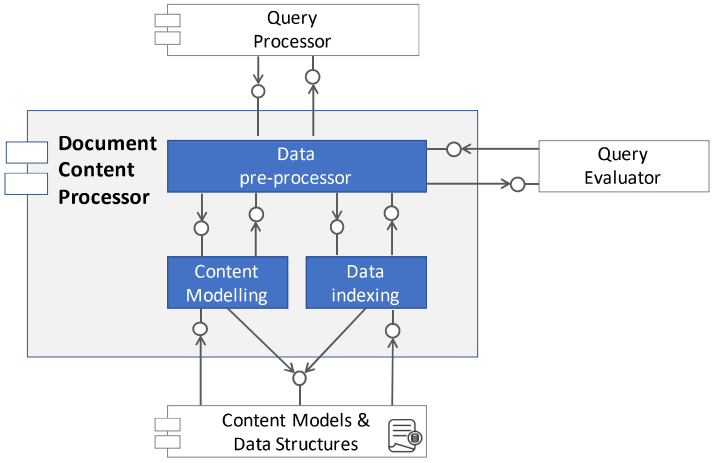

Figure 4.2: Document Content Processor Architecture (UML component diagram)

The Document Content Processor component receives as input (multi-lingual) textual documents collections, and queries to process them and generate data structures and models that will represent their content. It consists of three sub-components (see Figure 4.2):

- Data Pre-processor component that transforms raw data into a clean dataset and generates a vocabulary consisting of the terms used in the collection.
- Data Indexing component that creates an inverted index matrix that represents the content of the input data collection[1]. For each term in the vocabulary, the index stores the documents which contain that term (inverted index). In addition, the Document Indexing component builds a term frequency matrix that describes the frequency of terms that occur in the documents collection. In the frequency matrix, a column corresponds to a document, a row to a term, and a cell contains the number of times a term appears in a document.
- Content Modelling component that uses machine learning models for processing the textual content and modelling it. This component consists of 2 modules that generate models:

[1] An inverted index is a word-oriented mechanism for indexing a text collection to speed up the searching task. It enables agile access to the position within a document in which a term appears.

- Word2Vec Embedding module that implements Word2vec [94] to learn word embeddings[2]. Note that the word embedding representation is the one that best captures words contextual, lexical, and sentimental characteristics.
- MBERT Embedding module that uses the MBERT[3] multilingual tokeniser to generate tokens that BERT's embedding layer will further process. It combines it with other techniques like feature extraction, linear convolution and classification to produce a model of the collection content. A detailed description of this process has been used to explore texts seeking misinformation in our paper [95] (see Chapter 5).

4.1.2 Query Processor

HILDEX assumes that users express their information needs using Boolean keyword queries. The user query Q consists of m keywords $(k_1, ..., k_m)$ that can be combined with operators such as AND and OR.

The Query Processor component processes an initial query. Its current version relies on Whoosh[4] as a baseline search system to process the query and retrieve the initial data samples. Also, it uses different weighting model such as TF×IDF [96] and Okapi BM25F [97] for ranking the retrieved results.

4.1.3 Exploration Processor

Figure 4.3 shows an overview of the interactive data exploration process implemented by the Exploration Processor component. It consists of three sub-components: Scheduler, Results Ranking and Feedback Processor.

First, the Scheduler begins the exploration process by letting a user define a Boolean keyword query. Then, the query is processed and evaluated by the Query Processor and Query Evaluator components. Afterwards, the Results Generator generates the first set of items (i.e., textual documents) that match the query.

The items in the results are ranked by the Results Ranking component and delivered to the user who provides feedback about the result interacting with

[2]The underlying assumption of Word2Vec is that two words sharing similar contexts also share a similar meaning and vector representation in the model. For example, if terms A and B have respectively $< A >$ and $< B >$ embeddings, the distance between $< A >$ and $< B >$ represents a quantitative indicator of A and B's semantic relatedness.

[3]https://github.com/google-research/bert/blob/master/multilingual.md

[4]Whoosh is a full-text indexing and searching Python library that supports disjunctive and conjunctive queries. - https://whoosh.readthedocs.io/en/latest/index.html

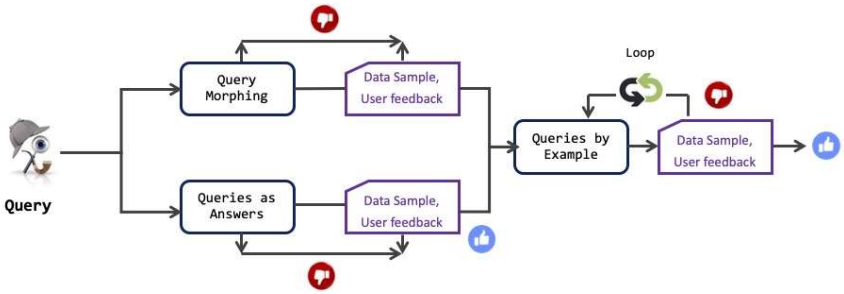

Figure 4.3: Overview of the exploration process.

the Feedback Processor component. This first result is used as a baseline to assess the results of further exploration tasks.

According to the user feedback, the Scheduler component triggers the process of proposing exploration techniques (e.g., query-morphing and/or queries-as-answers) that can be used for further exploring the data collection. These techniques provide new queries and data samples. Query ranking refers to ordering the generated queries according to their relevance to the given query under the users' exploration intent. The results produced by the Query Evaluator, Query Morpher, Query as Answers Generator and Query by Example Evaluator are ranked according to metrics (precision and similarity score).

The Feedback Processor assists the user in navigating the results space to find interesting queries and data samples. Accordingly, by exploring documents in results, explanations and metrics (e.g., precision, similarity), the user can decide to continue interacting with HILDEX for rewriting queries until she is satisfied with both queries and results (see figure 4.3).

4.1.4 Exploration Components

Query Morpher. This component reformulates an input query by using terms belonging to the results in the close area of the initial query results. The objective is to increase the possibility of exploring the most number of items in a collection. The Query Morpher component generates multiple morphs to help the user gain knowledge about data related to her submitted query. Query morphs query potentially obtain more results and assists the user in constructing exploratory queries.

Query-as-Answers Generator. This component produces rewritten queries by expanding the original query to rewrite a short and inaccurate query into a flawless query. The Query-as-Answers Generator component leverages three query expansion approaches to obtain better performance: synonym based,

knowledge-based and result set post-processing queries (see queries-as-answers in Chapter 3).

Query-by-Example Evaluator. Once the user is satisfied with most of the retrieved data objects, she can decide to drill down in the current search space and see more similar data objects. Query-by-example is defined to drill down and find similar data objects. In the exploration process, when the user is satisfied with query morphing or queries-as-answer, she is asked to choose relevant items among retrieved data. HILDEX utilizes the user's feedback and constructs a query-based on marked data samples. The constructed query help the user to find new data samples which are similar to marked data samples. HILDEX interacts with the user to get feedback about new data samples and starts the loop again until the queries proposal converges with user expectations.

4.2 Query exploration workflow

As shown in Figure 4.4, HILDEX implements a human-in-the-loop exploration workflow that coordinates the execution of a series of textual data exploration algorithms. An initial keyword query triggers this workflow. It applies techniques to analyse data collections and propose sets of queries that produce better precision and similarity scores. The user adjusts partial results, and at the end, the user obtains a set of queries and documents that fulfil her requirements.

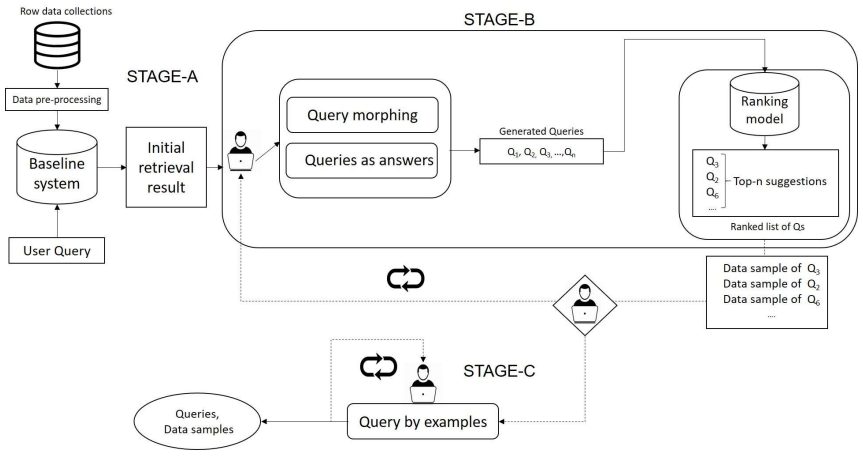

Figure 4.4: HILDEX exploration workflow.

Algorithm 4 presents the human-guided data exploration workflow consisting of three stages:

- STAGE-A: Given a textual content trained model (word2vec, BERT) and a Boolean query, this stage generates a vector representation using the inverted index and data indexes to look up records that match.
- STAGE-B: Given a first result on the initial query, the user can start an exploration process choosing iteratively one of the exploration operations proposed by HILDEX. The choice is made by analysing the set of generated queries by the operations and associated data samples. STAGE-B continues until the user finds her desired results.
- STAGE-C: The user tags the data samples she is interested in, and the query-by-example operation explores the collection looking for similar documents. HILDEX interacts with the user to get feedback about new data samples and starts the loop again until queries, and the data samples proposal, converge with user expectations.

4.2.1 Query generation

Query exploration first finds relevant documents matching and initial query, and then generates a set of queries using retrieved documents. This strategy a based on query generation reverse engineering (QGRE) [88, 98–100]. Given a corpus of documents D and a set of relevant documents ($D_{rel} \in D$), QGRE generates one or more queries $Q^{'}$ such that the result of the query $Q^{'}$ on data collection D (denoted by $Q^{'}(D)$) covers D_{rel}.

We used a decision binary tree-based method for generating Boolean queries [88] and implementing QGRE. As shown in Figure 4.5, using documents retrieved by an exploration operation (query morphing or query-by-examples), we train decision trees to generate a Boolean query [101–103] that can be matched with relevant documents.

In a binary decision tree, each internal node represents a feature (corresponding to a term appearing in a set of training documents). An internal node has two out-going edges labelled as True and False. Finally, the label of the leaf is either positive or negative, and it indicates the final decision.

Algorithm 5 implements the process of generating Boolean queries. Given a training dataset[5] and a set of candidate terms, the algorithm 5 produces a set of Boolean queries.

The intuitive idea behind any decision tree algorithm is as follows:

[5]The training set consists of the set of top-k documents matching the initial query (i.e., pseudo-relevant documents) and a set of non-relevant documents (documents beyond k).

Algorithm 4: HILDEX algorithm

Inputs: user query; row data collections: D; trained word embedding model: W;
Outputs: Ranked list of generated queries:Q along with data samples:S;
    Initialize $Q = \{\}$ $S = \{\}$
STAGE-A ;
    - data ← data preprocessing(D)
    - inverted index ← indexing(data)
    - vectorized dataset ← vector representation using trained word2vec model (data)
    - query vector ← vector representation (user query)
    - initial result ← using Whoosh to retrieve relevant results (user query, data)
STAGE-B ;
    - **while** user unsatisfied or wants to explore more:
        - user chooses one of the exploration techniques (query morphing or queries-as-answers)
        - **if** query morphing is selected:
            - Q, S ← Query Morphing (query vector, vectorized dataset)
        else:
            - Q, S ← Queries as Answers (user query, inverted index, data)
        - suggested queries, data samples ← Query Ranking(Q, S)
STAGE-C;
    - **while** user wants to see more similar results like some data samples:
        - examples ← user selects desired data samples from S
        - examples vector ← vector representation (examples)
        - Q, S ← Query by Examples(examples vector,vectorized dataset)
Return Q, S**.**

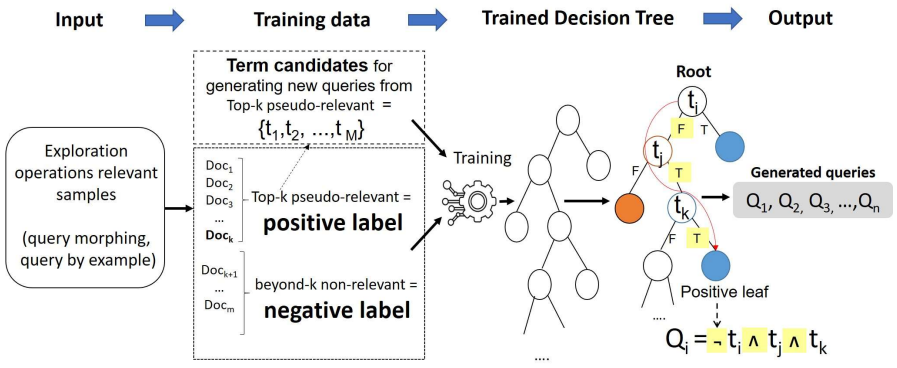

Figure 4.5: Boolean Query Generation.

Algorithm 5: Boolean Query Generation

Inputs: K different sets of query term candidates s.t. { $candidate_1$, $candidate_2$, ..., $candidate_K$ } where $candidate_i$ is a set of terms;
Training data s.t. {top k pseudo-relevant documents(positive labels) , k non-relevant documents (negative labels)}: B
Outputs: A set of Boolean queries: Q
$Q = \{\}$
for i = 1,..., K do:

- Train a decision tree using B and $candidate_i$ as query term candidates.
- Find paths from the root to every positive leaf in the decision tree and transform the rules (paths) into Boolean queries $\{q_1, q_2, ..., \} : BQS$.
- Add the Boolean queries BQS to Q.

Return Q.

1. Select the best feature using Attribute Selection Measures (ASM) to split the records.
2. Make that feature a decision node and breaks the dataset into smaller subsets.
3. Start building the tree by repeating this process recursively for each child until one of the conditions matches:
   - All the tuples belong to the same feature value.
   - There are no more remaining features (candidate query terms).
   - There are no more instances (training data).

To learn a binary decision tree, we need both training data and candidate query terms. In our approach, the training data are (i) the top-k relevant documents produced in the query exploration process considered positive labels, and (ii) non-relevant documents (ranked beyond-k) considered negative labels. Candidate query terms are frequent terms within pseudo-relevant documents (the documents that match the initial query) that are semantically close to the initial query.

Figure 4.6 illustrates how to generate Boolean queries using an example decision tree whose query term candidates are "building", "damage", and "historic".

For any input documents, from the root to a leaf node, we explore the tree to choose branches for which the condition holds. In our example 4.6, since the documents that matched the initial query and were used for training include the terms "damage" and "building", a document containing such terms is classified as relevant.

Accordingly, the path from the root to the first $True$ leaf can generate the query:

- $Q_1 : (building \wedge damaged)$, which is expected to retrieve documents containing damage and building.
- $Q_2 : (\neg building \wedge historic)$. Since we do not support queries with negation, only one word will remain in $Q_2 : (historic)$.
- $Q_3 : (damaged \wedge (home \vee historic))$. The algorithm replaces the most similar keyword in the original query with the one-keyword query Q_2. For the initial query $Q : (damaged \wedge (home \vee place))$ we substitute keyword "place" with the one-keyword query $(historic)$. We adopt this strategy to avoid reducing the initial query and looking at the effect of the Boolean query as proposed in [104].

4.2.2 Query ranking

HILDEX implements two approaches for ranking generated Boolean queries: similarity-based ranking and precision at k.

Similarity-based ranking Figure 4.7 illustrates the process of ranking a set of queries by similarity. The principle is to determine to which extent a generated query is similar to an (initial) query. The objective is to avoid query drifting [105] that happens when a large proportion of non-relevant documents are included as top-ranks.

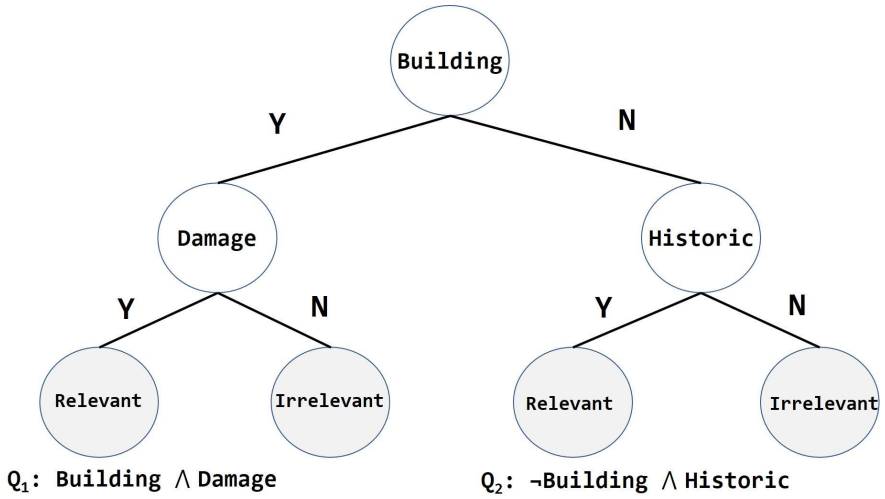

Figure 4.6: A tiny example of decision tree-based Boolean query generation.

As shown in figure 4.7, by using a trained word embedding model, we convert the original query and generated queries into a vector representation such that the queries that are closer in the vector space are expected to be similar in the context. After this step, we calculate the similarity score between the original query vector and all generated query vectors by using Cosine similarity. Finally, queries are sort in descending order w.r.t their similarity score.

Precision at k ranking Given a cut-off rank (k), P@k considers only the top-k results returned by the system for a query to evaluate it (e.g., P@20 or "Precision at 20" corresponds to the number of relevant results among the top 20 retrieved documents). As shown in figure 4.8, for the ranking of generated queries, we consider the top 50 retrieved results for each query and calculate its P@50. In the end, queries are sort in descending order of Precision at 50 value.

$$Precision = \frac{|number\ of\ relevant\ documents\ retrieved\ by\ a\ query|}{|total\ number\ of\ documents\ retrieved\ by\ that\ query|} \quad (4.1)$$

$$Recall = \frac{|number\ of\ relevant\ documents\ retrieved\ by\ a\ query|}{|total\ number\ of\ existing\ relevant\ documents|} \quad (4.2)$$

At the end of the ranking task, we have two sorted sets of queries. One

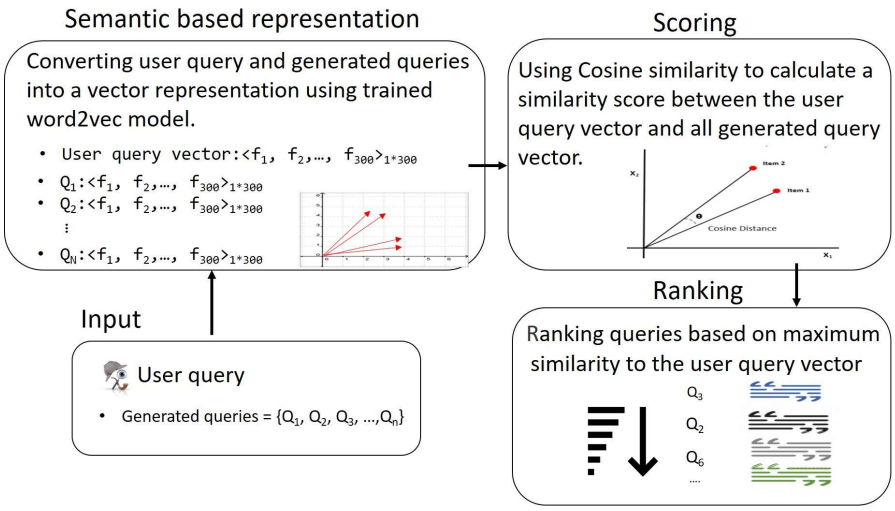

Figure 4.7: Ranking queries based on the maximum similarity with the user query vector.

set consists of top queries that are semantically close to the user query, and another set includes queries that have high precision. The user finally can review queries along with an explanation of why recommended query is interesting.

4.3 Using HILDEX

To illustrate our vision of data exploration with HILDEX, let us come back to the motivating scenario introduced in Chapter 1. Assume that a collection of micro-texts (tweets) exchanged by people during 8 crisis events has been pre-processed using the Document Content Processor.

According to the HILDEX exploration workflow (see figure 4.3), an exploration process for this example follows several stages that are triggered by an initial Boolean query expressed by a user. At each exploration stage, HILDEX generates queries and their corresponding results. The user can analyze the queries with additional data samples produced using the query-by-example operation.

In the first stage of the exploration workflow, instead of seeking an "optimal" query targeting relevant documents, query morphing and queries-as-answers operations return queries and their corresponding results. The user can analyze the rewritten queries with their associated data samples. If the result does not fulfil the user's expectation in terms of precision and similarity of relevant data, HILDEX assumes that the initial query has not

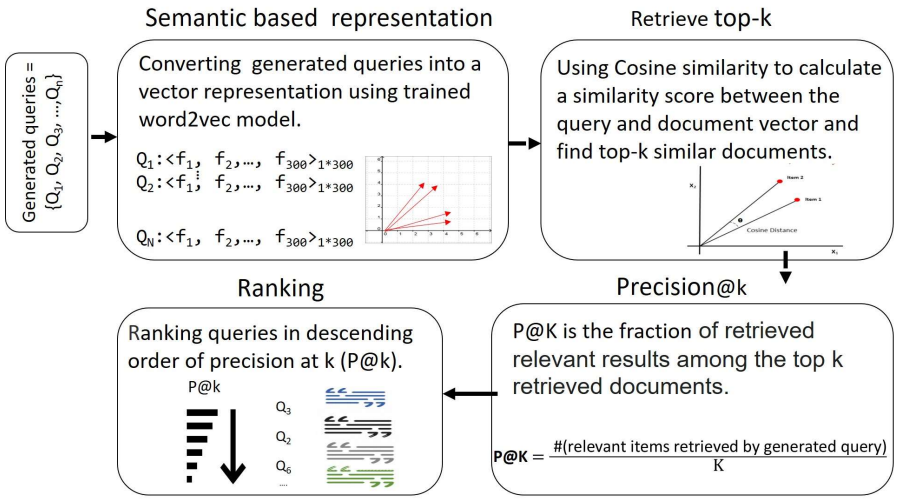

Figure 4.8: Ranking queries based on Precision@k.

been well defined and does not meet the user's expectations. In this case, the query must be refined.

Assume a user who is responsible for extracting useful information for disaster relief is looking for missing people and/or people who lost their homes and need shelter. As shown in figure 4.9, the user first poses query Q ("missing AND lost AND shelter"). HILDEX computes the query and produces the first set of results that seem not completely satisfactory for the user. So, after reviewing the retrieved results, the user decides to explore the data collection further. In this step, the user can choose one of the exploration operations (e.g., query morphing and/or queries-as-answers).

4.3.1 Query Morphing

Recall that the principle of query morphing is to extend the target area by including documents related to those in the initial result. If query morphing is selected, HILDEX proposes the following queries:

```
Q1: ("missing AND lost AND help")
Q2: ("help AND people AND find")
Q3: ("shelter AND food")
Q4: ...
```

These are morphed queries that HILDEX associates with data samples for letting the user explore and decide which of these queries produce relevant

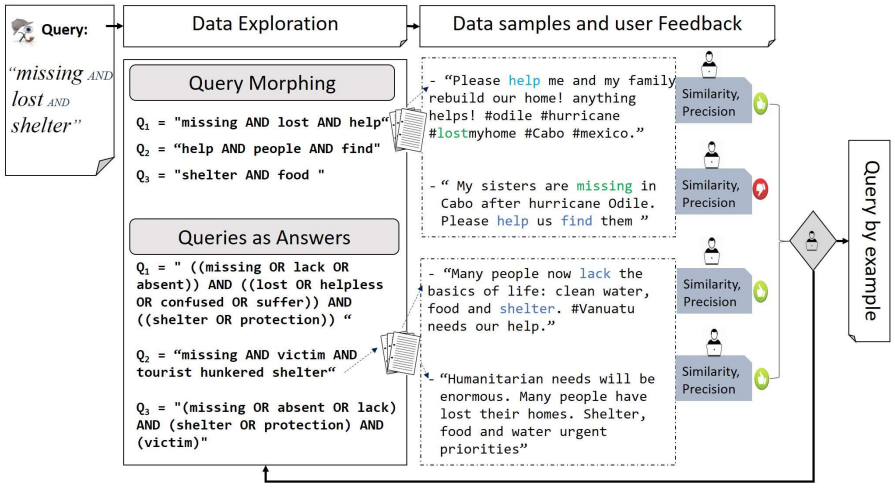

Figure 4.9: Exploration operations process.

results according to her expectations. For example, HILDEX can first propose a sample containing the following tweets:

```
"Please help me and my family rebuild our home!
anything helps!
#odile #hurricane #lostmyhome #Cabo #mexico"

"My sisters are missing in Cabo after hurricane Odile.
Please help us find them"
```

As shown above, terms such as "help", "find", "people", and "food" are new terms extracted from results in the close area of the initial query results that were used to generate new queries.

4.3.2 Queries-as-Answers

Queries-as-answers uses statistical and semantic approaches to obtain expansion terms. In the above example, terms like "lack" and "absent" are synonyms of "missing", and they are semantically related. However, the term "victim" frequently appeared in top k results, and it is statistically significant for rewriting the query. As shown in Figure 4.9, if the user chooses queries-as-answers instead of using query morphing, the suggested queries will be:

```
Q_1: "(missing OR lack OR absent) AND
      (lost OR helpless OR confused OR suffer) AND
```

```
        (shelter OR protection)"
    Q_2: (missing AND victim AND tourist hunkered shelter)"
    Q_3: "(missing OR absent OR lack) AND
        (shelter OR protection) AND (victim)"
    Q_4: ...
```

These data samples produced by these queries are:

```
"Many people now lack the basics of life: clean water,
food and shelter. #Vanuatu needs our help."

"humanitarian needs will be enormous.
Many people have lost their homes.
Shelter, food and water urgent priorities".
```

Generating results. Recall that HILDEX uses a decision tree technique to rewrite a query by morphing it or deriving queries. The question is how to choose "similar" or "related" terms and expressions that can be used for this purpose. In the decision tree of Figure 4.10, each internal node has five attributes.

The first attribute shows a decision rule that splits the data. If the condition holds, it means that the corresponding candidate term cannot generate the query. The *Gini* attribute (referred as Gini ratio [106]) measures the impurity of the node. A node is considered pure when all of its records belong to the same class. Such nodes are known as the leaf node.

Samples shows the size of training data and *values* represent the number of documents in each class (the left one shows the number of non-relevant class and the right one shows the size of the relevant class) and the last element illustrates that the majority of instances belong to relevant (True value) or non-relevant classes (False value).

To generate Boolean queries, HILDEX explores the tree to find all paths from a root to a positive leaf node (leaf with True class), and then we transform the rules (paths) into Boolean queries. Figure 4.10 shows an example of a decision tree generated by HILDEX in the query morphing and queries-as-answers operations. For example, the query term candidates used to construct the decision tree for a given query *(need and donation)* are {relief, emergency, earthquake, donate, fund, consider, money, effort }. As illustrated by the example 4.10, using the decision tree, according to the number of positive leaves (marked with Blue in the chart), 7 Boolean queries can be generated.

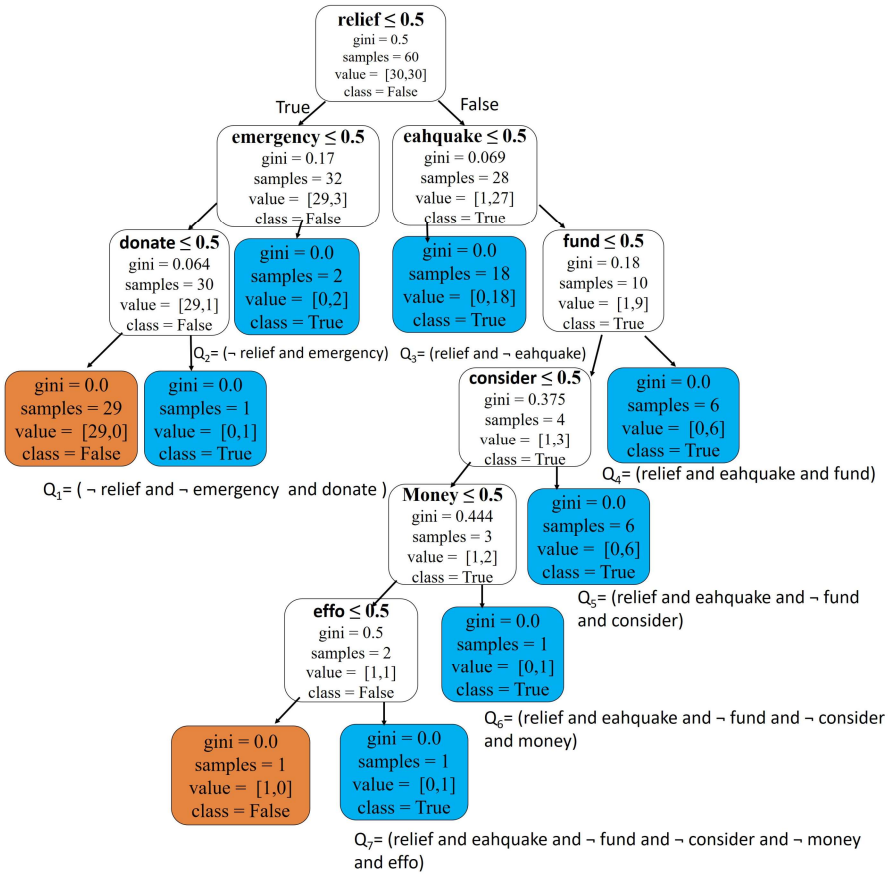

Figure 4.10: An example of constructed decision tree using query morphing.

Once the exploration results (rewritten queries and samples) fulfil the user expectations, the user might continue to explore data collections: select one of the generated queries and re-issue it to trigger a new iteration of the exploration process using the query example operation (see below).

4.3.3 Query-by-Example

Once the user is satisfied by the generated queries and data samples and feels good knowledge of documents in data collections, she can drill down on results and select desired results. By using query-by-example, HILDEX explores and finds documents that are similar to the selected one. In other words, user after specifying which subset of the data she is interested in, the HILDEX replies with similar results. As shown in figure 4.11, in our example, the user selects three data samples as relevant documents:

– $Example_1$: "Please help me and my family rebuild our home! anything helps! #odile #hurricane #lostmyhome #Cabo #mexico."

– $Example_2$: "Many people now lack the basics of life: clean water, food and shelter. #Vanuatu needs our help."

– $Example_3$: "Humanitarian needs will be enormous. Many people have lost their homes. Shelter, food and water urgent priorities"

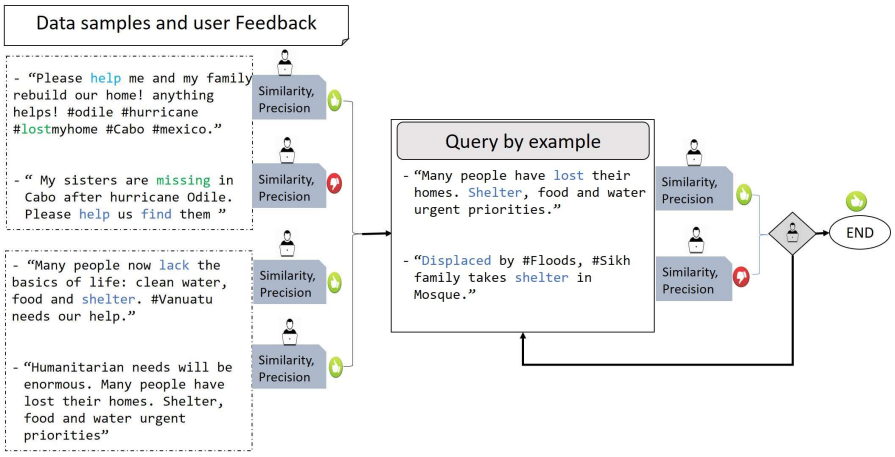

Figure 4.11: Exploration process.

Note that HILDEX uses the provided examples to deduce the user's query most likely intended and then finds similar results interesting for the user. In our example, retrieved data samples will be:

```
"Many people have lost their homes.
Shelter, food and water urgent priorities"

"Displaced by #Floods, #Sikh family
 takes shelter in Mosque".
```

Note that the context of retrieved results is wholly related to the given examples. At this stage, as shown on the right side of figure 4.3, HILDEX interacts with the user to get feedback about new data samples and starts the loop again until queries, and the data samples proposal converges with user expectations.

4.4 Conclusions

We proposed HILDEX, an textual data collections exploration framework that proposes a workflow that helps users to explore raw data collections and obtain insight into them interactively. HILDEX relies on information retrieval principles and machine learning-based Natural Language Processing (NLP) to process queries and data collections' textual content. Exploration is done by applying iteratively two query expansion techniques (query morphing and queries-as-answers) and a documents retrieval one (query-as-example). In the first stage of the HILDEX exploration workflow, the objective is to propose queries that can give the best insight into the collection content for an initial query. This initial (Boolean) query represents a user's first exploration intent. To decide which queries potentially produce relevant results, HILDEX produces results by associating queries with content samples. Thus, the results consist of tuples of the form $< query, sample >$ that the user analyses. The user tags those queries that produce results close to her expectations and can run such queries under a query by example fashion. She obtains documents similar to those that seem relevant. The exploration process is iterative and converges when the user is satisfied with the results.

In existing work, there are no standard specifications for data exploration techniques. We proposed algorithms adapted to rewrite (expand) Boolean keyword queries that explore textual data collections. These algorithms are thoroughly described in Chapter 4, and they are profiled using experiments that validate them using performance scores. These algorithms have been wrapped as operators in the first version of the implementation of HILDEX. Chapter 5 describes examples used for validating HILDEX.

CHAPTER 5

Validating HILDEX through Exploration Use Cases

The target audience of HILDEX is users unfamiliar with the structure and content of the data collections they are willing to explore. Furthermore, these users have vague ideas of the type of questions they can ask for exploiting the content and fulfilling their information requirements. So they can have difficulty in formulating their informational needs as queries. This chapter describes three use cases proposing different exploration scenarios that were addressed with HILDEX. The objective is to validate the exploration workflow implemented by HILDEX and show how certain workflow phases are more important than others, depending on the type of exploration to perform. For the experimentation, we used the Google Colaboratory cloud platform[1] with 25GB NVIDIA GPU RAM, Xeon Processors @2.3Ghz and CUDA-10.0.

The chapter is organized as follows. Section 5.1 describes the data collections associated with the use cases we defined for validating HILDEX. It also describes the pre-processing tasks performed for associating them to the HILDEX document content database. Section 5.2 introduces an exploration scenario intended to produce a classified overview of events described by micro-texts in a disaster management scenario, where volunteers and victims need to know what has happened (event), and what has been done to help/solve the situation. Section 5.3 describes a scenario for exploring scientific articles (i.e., text data collections) for answering a batch of queries

[1] https://colab.research.google.com

intended to understand the COVID-19. Section 5.4 describes a scenario where a classification and quantitative requirement guides the exploration of multi-lingual micro-texts to understand how misinformation about COVID-19 spreads across different countries. Finally, Section 5.5 concludes the chapter.

5.1 Data collections

The use cases for validating HILDEX were based on three textual data collections that were pre-processed by the Document Content Processor (see Section 4.1 HILDEX general architecture). The crisis data collections were processed using a model adapted for crisis terms (i.e., vocabulary) and a classifier for further classifying the micro-texts into *events* (happening during a crisis situation) and *actions* performed in response to events. The data collection about COVID-19 misinformation contained multi-lingual micro-texts, thus we enhanced HILDEX with other machine learning methods, namely (M)BERT and other BERT models adapted to deal with specific languages to pre-process their content. The COVID-19 scientific papers collection implied creating a domain vocabulary related to the disease. The final version of the HILDEX Document Content Processor provides now several text-processing tools that are even calibrated for addressing text on target knowledge domains and language.

The data collections used for validation were indexed by HILDEX using Whoosh[2] and ranked using the Okapi BM25F[3] ranking function[4].

5.1.1 Crisis related tweets dataset

The CrisisNLP data collection [107] contains 35648 labelled data related to the crisis tweets posted during 8 disaster events (see table 5.1).

Dataset pre-processing. The dataset was cleaned by removing duplicates and near-duplicates resulting into a total of 10044 tweets. The clean tweets collection was labelled first into various informative classes. Since the crisis management can be understood with respect to the events that come up during crisis and the actions performed to react to such events, the tweets were further classified into two classes: *Event* and *Action* (represents reactions to events). Table 5.2 shows the number of tweets for each class.

[2] https://whoosh.readthedocs.io/en/latest/intro.html - Accessed the 14th July 2021

[4] Okapi BM25 (BM is an abbreviation of best matching) is a ranking function used by search engines to estimate the relevance of documents to a given search query https://en.wikipedia.org/wiki/Okapi_BM25 - Accessed the 14th July 2021

Table 5.1: Summary of the CrisisNLP dataset per event

Crisis Name	Country	Number of Tweets	Year
Nepal Earthquake	Nepal	12489	2015
Typhoon Hagupit	Phillippines	9675	2014
Cyclone PAM	Vanuatu	2613	2014
Chile Earthquake	Chile	2453	2014
Hurricane Odile	Mexico	2196	2014
California Earthquake	USA	2196	2014
Pakistan Floods	Pakistan	2013	2014
Pakistan Earthquake	Pakistan	2013	2013
Total Number		35648	

Table 5.2: Crisis Event and Action Tweets dataset

Class	Total label	Description
Event	3842	Tweets reporting occurrence and happening of events during the crisis. Reports deaths, injuries, missing, found, or displaced people,infrastructure and utilities damage
Action	6202	Tweets reporting responses and measures taken by people during crisis. Messages containing donations or volunteering offers also sympathy-emotional support
Total number	10044	

5.1.2 Misinformation dataset

The misinformation dataset is a multiligual set of micro-texts about COVID-19, collected from different sources that provided raw texts and labelled ones. We built two data collections, the first intended to train HILDEX content processing models, and the second for testing them.

Training dataset. We collected micro-texts from an online fact-checker website called Poynter [108]. Poynter has a specific COVID-19 related misinformation detection program named "CoronaVirusFacts/DatosCoronaVirus Alliance Database".[5] This database contains thousands of labelled social information such as news, posts, claims, articles about COVID-19, which were manually verified and annotated by human volunteers (i.e., fact-checkers) from all around the globe. The database gathers all the misinformation related to COVID-19 like cure, detection, the effect on animals, foods, travel, government policies, crime or lockdown. The misinformation dataset is available in English and Spanish and accessible via a search engine[6].

We web scrapped the search engine results using Beautifulsoup[7], a Python library for scraping information from web pages, and collected 8471 English micro-texts containing false news/information labelled according to nine misinformation classes: *False*, *Partially False*, *Misleading*, *No evidence*,

[6] https://www.poynter.org/covid-19-poynter-resources/

Incorrect, *Four Pinocchios*[8], *Three Pinocchios*[9], *Two Pinocchios*[10] and *Mostly False*. We also gathered the article's title, the content, and the fact checker's misinformation-type label for each article[11].

For Spanish, we used another Latin American website specialized in fact-checking called "Chequeando".[12] We collected 531 misinformation articles containing the misinformation published on social media platforms such as Facebook, Twitter, Whatsapp, YouTube. Micro-texts were mainly related to political-biased news, scientifically dubious information and conspiracy theories, misleading news and rumours about COVID-19[13].

We also used a human-annotated fact-checked tweet dataset [109] available at a public repository[14]. The dataset contained *true* and *false* labelled micro-texts (i.e., tweets) in English and Arabic. We used only 500 English tweets labelled as false. We compiled a total of 9,502 micro-texts distributed across 9 misinformation classes shown in Table 5.3.

Table 5.3: Training misinformation dataset summary

Classes	Language	Number of tweets
False [108]	English	2,869
False [109]	English	500
False	Spanish	191
Partially False	English	2,765
Partially False	Spanish	161
Misleading	English	2,837
Misleading	Spanish	179
Total		9,502

Testing dataset. We further collected around 2,137,106 multilingual tweets. The tweets were expressed in eight languages: English, Spanish, Indonesian, French, Japanese, Thai, Hindi and German. Therefore, we used a dataset of tweets IDs associated with the novel coronavirus COVID-19 [110]. Starting on January 28, 2020, the current dataset contains 212,978,935 tweets divided into groups based on their publishing month. The dataset was collected using multilingual COVID-19 related keywords and contained tweets in more than 30 languages[15]. We decided to retrieve the tweets using the tweet IDs published during five months in 2020 (February, March, April,

[8] 90%-95% changes of it being false
[9] 70%-75% changes of it being false
[10] 50%-55% changes of it being false
[11] https://www.crummy.com/software/BeautifulSoup/bs4/doc/
[13] https://chequeado.com/latamcoronavirus/
[14] https://github.com/firojalam/COVID-19-tweets-for-check-worthiness
[15] We used tweepy, a Python library for interacting with the Twitter API http://www.tweepy.org

May and June). Table 5.4 shows the total number of collected tweets. The distribution of tweets across eight languages corresponds to most English items (almost 1 and 1/8 of the whole dataset), then Spanish (1/4 of the total number of tweets) and the rest for French, Japanese, Indonesian, Thai, and Hindi.

Table 5.4: Testing misinformation dataset summary

Language	ISO	Number of tweets
English	en	1,472,448
Spanish	es	353,294
Indonesian	in	80,764
French	fr	71,722
Japanese	ja	71,418
Thai	th	36,824
Hindi	hi	27,320
German	de	23,316
Total		2137106

Data Collections Cleaning. The data collections contained noise such as emojis, symbols, numeric values, hyperlinks to websites, and username mentions that were removed in the cleaning phase. Since the dataset is multilingual, cleaning had to be calibrated to avoid not loosing valuable information. We included in HILDEX the possibility of using regular expressions to remove URLs, special characters or symbols, blank rows, re-tweets, user mentions. The hashtags were not removed because they might contain helpful information. For example, in the sentence *"Wear mask to protect yourself from #COVID-19 #corona"*, only the symbol # was removed. HILDEX relies on NLTK[16] for removing stop words. NLTK supports multiple languages except for few languages, such as Hindi and Thai. For cleaning the Hindi dataset, HILDEX uses CLTK (Classical Language Toolkit)[17]. For removing Thai stop words from Thai tweets, HILDEX uses PyThaiNLP [111]. The emojis were removed using their Unicode.

Attribute engineering The collected data was unevenly distributed across nine classes: *No evidence*, *Four Pinocchios*, *Incorrect*, *Three Pinocchios*, *Two Pinocchios* and *Mostly False* (the smallest group). Most collected articles were labelled either as *False*, *Partially False* and *Misleading*. We performed an attribute engineering phase for preparing the dataset, produced a uniformly distributed dataset and reorganised the initial dataset as follows.

[16]NLTK https://www.nltk.org/ is a Python library for natural language processing.
[17]https://docs.cltk.org/en/latest/index.html

The classes *Four Pinocchios* and *Incorrect* were merged with the class *False*. The classes *Three Pinocchios* and *Two Pinocchios* were merged into the class *Partially False*. The classes *No evidence* and *Mostly False* were merged with the class *Misleading*.

Data Collections Pre-processing. The objective was to produce a labelled multi-lingual dataset. Therefore, pre-processing multilingual data collections consisted in the following tasks (see figure 5.1): tokenizing, text features extraction, linear transformation, and classification. The first phases (tokenizing, text feature extraction, linear transformation) correspond to a substantial data-preparation process intended to build a multi-lingual vectorized representation of texts. The objective is to achieve a numerical pivot representation of texts agnostic of the language. The classification task uses a dense layer and leads to a trained network model that can be used to classify micro-texts (e.g. tweets) into three misinformation classes: *false*, *partly false* and *misleading*.

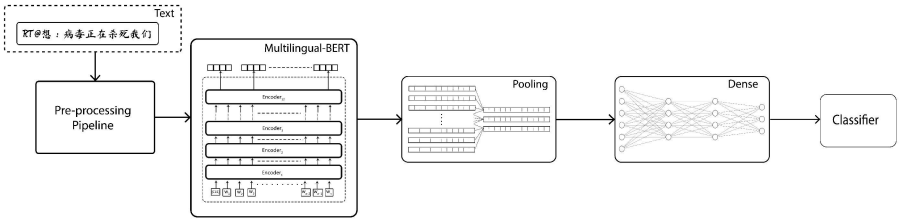

Figure 5.1: Multi-lingual Data Collections Pre-Processing Pipeline

- ***Data preparation process***: the objective is to build a vectorized representation of the multi-lingual data collections. It is done through three tasks:

  - *Text tokenization* uses the BERT multilingual tokeniser to generate tokens that BERT's embedding layer will further process. HILDEX uses MBERT[18] to extract contextual features, namely word and sentence embedding vectors, from text data. In the subsequent phases that use NLP models, these vectors were used as feature inputs with several advantages. (M)BERT embeddings are word representations dynamically informed by the words around them, meaning that the same word's embeddings will change in

[18] https://github.com/google-research/bert/blob/master/multilingual.md

(M)BERT depending on its related words within two different sentences.

For the non-expert reader, the tokenization process is based on a WordPiece model. It greedily creates a fixed-size vocabulary of individual characters, subwords, and words that best fit a language data (e.g. English)[19]. Each token in a tokenized text must be associated with the sentence's index: sentence 0 (a series of 0s) or sentence 1 (a series of 1s). After breaking the text into tokens, a sentence must be converted from a list of strings to a list of vocabulary indices.

The tokenization result is used as input to apply BERT that produces two outputs, one pooled output with contextual embeddings and hidden states of each layer. The complete set of hidden states for this model are stored in a structure containing four elements: the layer number (13 layers)[20], the batch number (number of sentences submitted to the model), the word / token number in a sentence, the hidden unit/feature number (768 features)[21]. For the preparation of the data collection for HILDEX tokenization is more complex because it is done for sentences written in different languages. Therefore, it relies on the MBERT model that has been trained for this purpose.

- *Feature Extraction* is intended to exploit the information of hidden-layers produced due to applying BERT to the tokenisation phase result. The objective is to get individual vectors for each token and convert them into a single vector representation of the whole sentence. For each token of our input, we have 13 separate vectors, each of length 768. Thus, to get the individual vectors, it is necessary to combine some of the layer vectors. The challenge is to determine which layer or combination of layers provides the best representation.
- *Linear convolution.* The hidden states from the 12th layer are processed in this phase, applying linear convolution and pooling to get correlation among tokens. We apply a three-layer 1D convolution over the hidden states with consecutive pooling layers.

[19]This vocabulary contains whole words, subwords occurring at the front of a word or in isolation (e.g., "em" as in the word "embeddings" is assigned the same vector as the standalone sequence of characters "em" as in "go get em"), subwords not at the front of a word, which are preceded by '##' to denote this case. Individual characters [112]

[20]It is 13 because the first element is the input embeddings, the rest is the outputs of each of BERT's 12 layers.

[21]That is 219,648 unique values to represent one sentence!

The final convolutional layer's output is passed through a global average pooling layer to get a final sentence representation. This representation holds the relation between contextual embeddings of individual tokens in the sentence.

– **Classification:** The objective is to produce a labelled data collections where tweets are labelled according to three classes: *false*, *partly false* and *misleading*. This classification layer outputs a Softmax value of vector, depending on the output. The index of the highest value in the vector represents the label for the given sequence: *false*, *partly false* and *misleading*.

Multi-lingual Misinformation Tweets Data Collection. The result is a labeled tweets data collection stated in different languages. Table 5.5 gives an overview of the dataset and showcases some misinformation articles. Column 2 shows the original label assigned by the fact-checker, column 3 gives a misinformation example associated with the label present in column 2, and column 4 provides reasoning given by the fact-checker behind assigning a particular label (column 2) to the misinformation (column 3). Finally, column 1 (see table 5.5) corresponds to the label assigned during the data pre-processing phase.

For example, if we look at the entry number '3' in the table 5.5, the misinformation is about the adverse effect of 5G radiation over the COVID-19 patients. This entry was labelled 'Incorrect' by the fact-checker. After analysing the fact-checker rating and the explanation, we labelled it as 'False' misinformation. Entry number '5' talks about the COVID-19 test cost. The explanation given by the fact-checker is valid as it is not sure if there is any fee in the USA for the COVID-19 test or not. So because of the lack of evidence and uncertainty, we labelled it as 'Partially false'. Entry number '7' in the table talks about a video showing COVID-19 corpus dumping in the sea. Based on the explanation, the video was coupled with the wrong information to mislead the audience. So it was labelled as 'Misleading' misinformation.

5.1.3 COVID-19 scientific articles

The CORD-19 [113] (COVID-19 Open Research Dataset) dataset is a corpus that currently includes over 100,000 scholarly articles and updates weakly. The Allen Institute for AI published this dataset for the global research community. The objective was to let the community apply natural language processing, natural language understanding and Artificial Intelligence tech-

Table 5.5: Misinformation Dataset

Our Rating	IFCN(Poynter) Rating	Misinformation	Explanation
False	False	The border between France and Belgium will be closed.	French and Belgian authorities denied it.
	Four pinocchios	Trump's effort to blame Obama for sluggish coronavirus testing.	There was no "Obama rule," just draft guidance that never took effect and was withdrawn before President Trump took office.
	Inaccurate	Elisa Granato, the first volunteer in the first Europe human tria of a COVID-19 vaccine, has died.	Elisa Granato, the first volunteer in the first Europe human trial of a COVID-19 vaccine, has died.
Partially False	Partially False	Media shows a Florida beach full of people while it's empty.	The different videos were not shot at the same time. The beaches are empty when they are closed.
	Two Pinocchios	The bill for a coronavirus test in the US is $3.000	The CDC is not making people pay the test by now.
	Partly False	Salty and sour foods cause the "body of the COVID-19 virus" to explode and dissolve.	"Consuming fruit juices or gargling with warm water and salt does not protect or kill COVID-19," the World Health Organization Philippines told VERA Files.
Misleading	Misleading	A clip from Mexico depicts the dumping of coronavirus patients corpses into the sea.	Misbar's investigation of the video revealed that it does not depict the dumping of coronavirus patients corpses in Mexico, but rather paratroopers landing from a Russian MI 26 helicopter.
	No Evidence	Media uses photos of puppets on patient stretchers to scare the public.	There is no evidence that any media outlet used this photo for their reporting about COVID-19. Its origin is unclear, maybe it was shot in Mexico and shows a medical training session.
	Mostly False	Coronavirus does not affect people with 'O+' blood type.	The post claiming coronavirus does not affect people with 'O+' blood type is misleading.

niques to generate new insights about COVID-19 and corona virus-related research (e.g., SARS and MERS).

CORD-19 is a collection of research papers published by various international publishing bodies. The papers' sources come from the databases PMC, bioRxiv, medRxiv, Elsevier, Springer Nature and WHO. The metadata involves papers published by bioRxiv, medRxiv, PMC, WHO and individual publishers. The data was provided in JSON format which we converted into CSV files with the schema columns: *paper_id title, authors affiliations, abstract, text, bibliography, raw_authors* and *raw_bibliography.* We further pre-processed the data to produce a clean and structured dataset.

We used the version-52 of the original CORD-19 [113] dataset which contained around 110,427 COVID-19 related research papers. Table 5.6 shows the CORD-19 data set details. The dataset schema consisted of the following attributes: *paper_id*, *body_text*, *methods*, *results*, *cord_uid*, *source*, *title*, *doi*, *pmcid*, *pubmed_id*, *license*, *abstract*, *publish_time*, *authors*, *journal*, *mag_id*, *who_covidence_id*, *arxiv_id*, *pdf_json_files*, *pmc_json_files*, *url*, *s2_id*, *is_covid19* and *publish_year.*

Table 5.6: The COVID-19 data set content.

Type of Article	Number of Articles
CORD-19	110,427
Papers without title	11515
Papers without abstract	19431
Full text papers	**90,767**

Data collection Pre-Processing is performed by first processing and indexing the papers collection to discover a set of topics and then assign relevant topics to the paper according to its content.

As discussed in chapter 3, HILDEX first pre-processes documents to generate vector representations for words (i.e., word embedding)[22]. Once texts had been pre-possessed, we enhanced the Document Content Processor with `Gensim Phrases` Python library [114] to automatically detect common phrases (bigrams) from each paper in the corpus. For example, sentences like *infectious disease* or *public health* must occur together. We applied the `skip-gram` method to predict the context of words. The principle of the `skip-gram` method is the use of a word for predicting its target context.

We trained a `Word2vec` model [94] to calibrate HILDEX to deal with this particular multi-lingual data collection[23]. We set the dimensionality of the feature vectors to 300 and trained the model for 15 epochs. Finally, the resulting `Word2Vec` model was stored by the Document Content Processor of HILDEX for future exploration tasks.

Data Engineering We did a first filtering process choosing 8 features from the dataset schema that represented papers content, i.e., *paper_id*: unique paper id for each article, *title*: title of the paper, *abstract*: abstract of the article, *body_text*: full length text of the article, *doi*: DOI id of the article, *url*: hyperlink to the article's publication website *publish_year*: publication year of the article and *is_covid19*: contains either true (COVID-19 only article) or false value. Table 5.7 shows the features of dataset schema.

We went further into an attribute engineering process to add an attribute named *complete text* to the schema. This new attribute concatenates three attributes of the dataset schema: *title*, *abstract* and *body_text*. In that way, we simplified the schema. We removed those research papers which were not

[22]The word embedding representation is the one that best captures words contextual, lexical, and sentimental characteristics.

[23]The model is a two-layer neural net that processes text by "vectorizing" the words in the document to build a vocabulary.

Table 5.7: The COVID-19 data set schema.

Feature	Description
paper_id	Unique paper id for each article
title	Title of the paper
abstract	Abstract of the article
body text	Full length text of the article
doi	DOI id of the article
url	Hyperlink to the article's publication website
publish year	Publication year of the article
is_covid19	Contains either true (COVID-19 only article) or false value.

written in English. We also removed the abstract-only papers.[24] Finally, we end up with a collection of 80,000 COVID-19 research papers.

We then performed the text preprocessing on the *complete text*. We used `ScispaCy` [116], a Python package containing spaCy models for processing biomedical, scientific or clinical text. We removed the stop words and performed word lemmatization on the text data. We also removed some common unnecessary words such as author, figure, copyrights, license, fig from the *complete text*. However, we kept important information such as citation numbers in papers intact to get a complete answer without missing values. We used *complete text* to train `word2vec` model and to extract sentences from each paper.

Generating a vocabulary The COVID-19 dataset consists of many scientific papers containing text in words, sentences and paragraphs. It is necessary to generate a content model to explore the documents (i.e., papers). In our approach, this model is a vocabulary. The vocabulary is particularly useful in the case of the COVID-19 because there is, for the time being, no official vocabulary about this topic.

Topic modelling We extracted the COVID-19 dataset topics to represent the content which consists of unstructured texts. We enhanced HILDEX with a statistical process for learning and extracting topics from documents. We adopted the topic modelling[25] method called Latent Dirichlet allocation (LDA) [117]. It is a Bayesian model for classifying discrete data having uncorrelated topics. We represented each scientific document in the corpus through topic modelling to distribute topics described as a distribution of words. LDA automatically analyses a text for clustering the words from a

[24] Research [115] has proven that using full text for information retrieval is more effective than just using the abstract section of the research paper.

given set of documents as an unsupervised machine learning method[26].

The number of topics plays a crucial role in deciding the performance of the model computed with LDA. After several iterations, we discovered that 50 is the optimum number of latent semantic topics extracted from the COVID-19 literature corpus. For producing fine-grained results, the discovered topics were specified and refined with minimum overlapping. Once LDA topic modeling is applied to COVID-19 literature corpus, the words that make up each hidden topic are extracted. Table A.8 in Appendix A.2 illustrates the top 10 words that belong to 50 topics after applying LDA to COVID-19 literature corpus. The words in each topic represent/describe the content of that topic. Then, having a set of topics assigned to each paper (i.e., a distribution over words), we organised papers in an LDA space, namely a simplex. The dimensionality of the space depends on the number of topics. Depending upon a topic's words distribution in a paper, each paper in the corpus is closer to the topics representing it more strongly.

COVID-19 Labelled Scientific Articles The result of this pre-processing task is a probability based multi labelled data set. Using LDA each document gets a probability of belonging to a specific topic for all topics. We tagged a document with just one label(topic), based on maximum of probability of the document belonging to a topic. Table 5.8 shows the number of papers associated to each topic. An article belong to a topic if it is closer to that topic rather than other topics.

5.2 Interactive exploration of disaster micro-texts

In the disaster management scenario (e.g., earthquake, flooding, fire) introduced as motivation in Chapter 1, social media data describing victims and volunteers situation and observations are explored for making critical decisions.

Use case objective. HILDEX is used to assist in expressing queries that can potentially explore disaster data collections and contribute to building situational awareness of the Emergency Operations Centers (EOCs) team to make strategic decisions.

For the disaster management scenario, we implemented an adapted interface using python scripts with ipywidgets[27] and Jupyter notebooks. The

[26]Through topic modelling, one can identify the topics that best describe a set of documents. Knowing the topics representing the content of documents can be helpful for search engines and customer service automation.

Table 5.8: Number of articles in each topic based of maximum probability.

Topic	Number of articles	Topic	Number of articles	Topic	Number of articles
Topic 1	5292	Topic 18	1332	Topic 35	645
Topic 2	1188	Topic 19	2460	Topic 36	1322
Topic 3	8312	Topic 20	232	Topic 37	23
Topic 4	242	Topic 21	1223	Topic 38	1446
Topic 5	2417	Topic 22	1087	Topic 39	706
Topic 6	904	Topic 23	4524	Topic 40	1943
Topic 7	499	Topic 24	1403	Topic 41	1239
Topic 8	1047	Topic 25	1860	Topic 42	1831
Topic 9	660	Topic 26	453	Topic 43	298
Topic 10	1012	Topic 27	659	Topic 44	2352
Topic 11	4492	Topic 28	1043	Topic 45	1564
Topic 12	479	Topic 29	18	Topic 46	46
Topic 13	897	Topic 30	1395	Topic 47	2569
Topic 14	300	Topic 31	6915	Topic 48	1189
Topic 15	1501	Topic 32	1823	Topic 49	1038
Topic 16	392	Topic 33	2572	Topic 50	2041
Topic 17	857	Topic 34	258		

HILDEX interface enables the exploration of crisis tweets and the selection of exploration operations and feedback expression for tagging desired data samples.

5.2.1 Profiling crisis evolution through social networks

Consider a user seeking for detailed information regarding infrastructure and utility damage in a crisis related tweet dataset. She first defines a query: 'damage and building'. As shown at the top of figure 5.2, HILDEX provides two sliders that allow the user to change an integer value within a set bounds. Thereby, the user can adjust the number of queries and the number of data samples that she wants to see in the slider results. HILDEX retrieves the initial related tweets and shows the top 10 retrieved tweets related to the user query (see the bottom of the figure 5.2). The user can start an exploration process choosing iteratively one of the explorations (i.e., query-morphing and/or queries-as-answers).

Assume that the user clicked on the query-morphing button. HILDEX provides new morphed queries and related tweets to assist the user navigating in the data space to find interesting queries and data samples. As shown in figure 5.3, when user clicked on the query-morphing button the query is morphed as follows:

```
Q_1 = "magnitude AND major"
Q_2 = "damage AND  building OR (damage AND building AND hit)"
```

[27]ipywidgets are interactive HTML widgets for Jupyter notebooks and the IPython kernel. https://ipywidgets.readthedocs.io/en/latest/

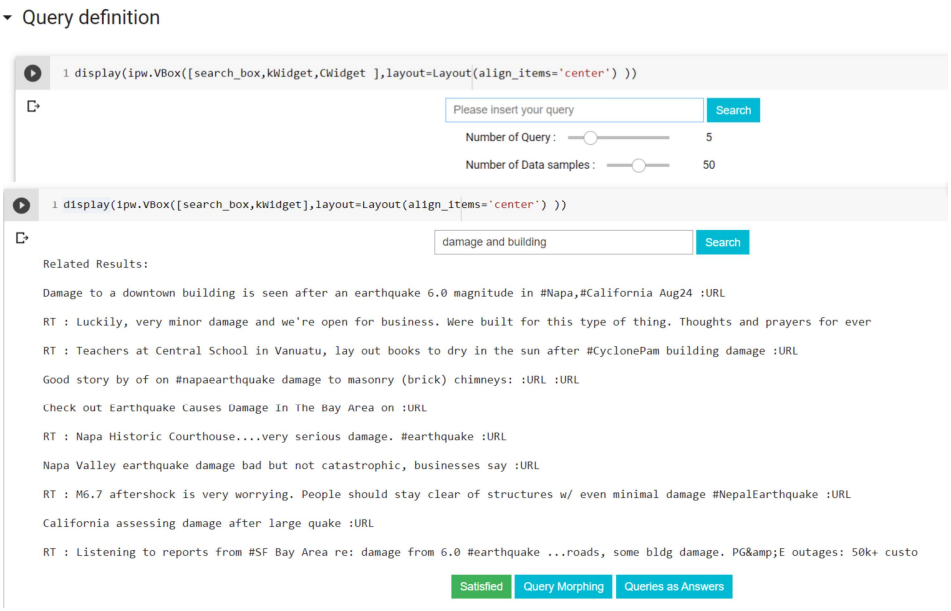

Figure 5.2: Code Snippet of HILDEX interface for exploring top-k results.

```
Q_3 = "damage AND  building AND caused"
```

At some step of the exploration, the user can tag the tweets she is interested in and use query-by-example to look for similar tweets. Figure 5.4 shows the situation in which the user marked four tweets as desired data samples:

```
{"Damage to a downtown building is seen after an earthquake
  6.0 magnitude in #Napa,#California ",
"Napa Historic Courthouse..very serious damage. #earthquake",
"California assessing damage after large quake",
"Listening to reports from #SF Bay Area re: damage from 6.0
#earthquake..roads, some bldg damage."}.
```

Input examples for query-by-examples exploration are collected using the checkboxes. After collecting all feedbacks, HILDEX explores and finds tweets that are similar to the selected ones (see figure). As shown in figure 5.5 the top 5 results of the query by examples operation is as follows:

```
{"Luckily, very minor damage and we’re open for business.Were
built for this type of thing. Thoughts and prayers for ever",
"California Quake Means Big Damage For Napa Valley Wineries:
```

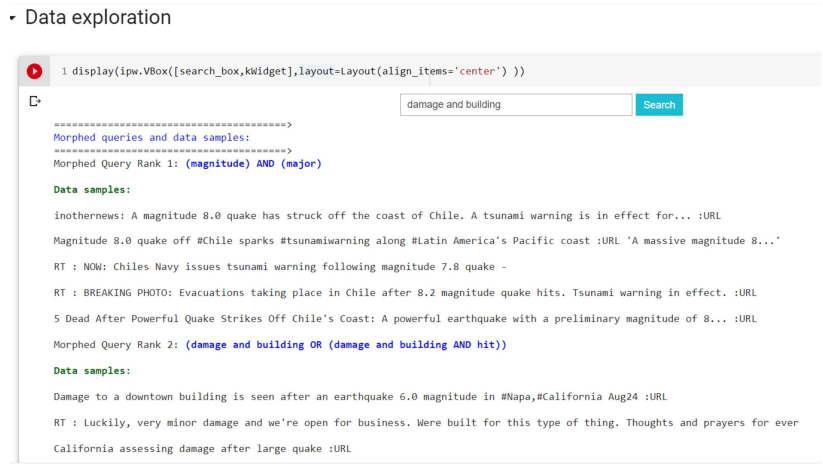

Figure 5.3: Code Snippet of HILDEX interface for performing query morphing.

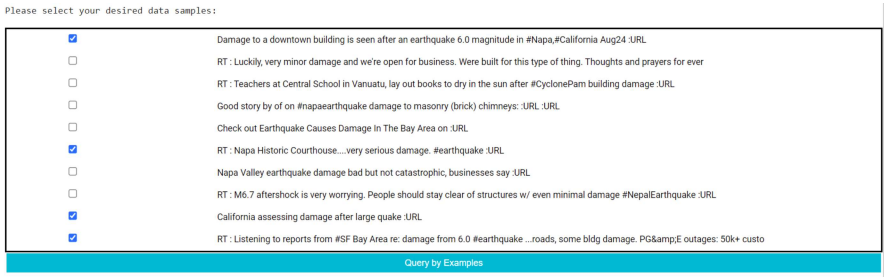

Figure 5.4: Code Snippet of HILDEX interface for selecting examples.

```
It's still too early to tell just how much the 6.0.",
"Damage to US Post Office bldg on Second St. Napa. Homeless
sleeping inside were hit with debris",
"Applause RT : California in one picture. MT : Napa skaters
find upside to quake damage",
"Pretty severe tank damage ;significant amount of #wine loss,
but we're feeling fortunate everyone is safe #napaquake."}.
```

Observe that the context of retrieved tweets is wholly related to the given examples. HILDEX interacts with the user to get feedback about new data samples and starts the loop again until queries, and the tweets proposal converges with user expectations.

Synthetic queries. For a more intensive experiment, we created synthetic queries using the vocabularies of the Event class (event-query class) and

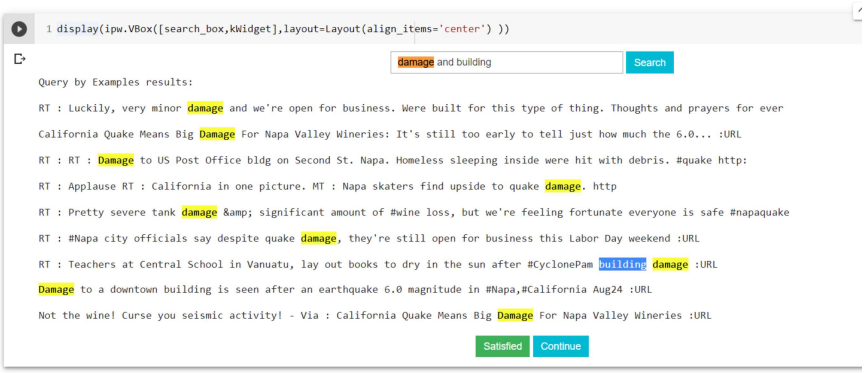

Figure 5.5: Code Snippet of HILDEX interface for performing query-by-example

Action class (action-query class). Table 5.9 illustrates 15 terms that frequently occurred in the Event and Action classes. We consider them as seed terms to construct synthetic queries. We have tested HILDEX using 1150 synthetic queries with sizes one, two and three.

Table 5.9: Candidate terms of each class for building synthetic queries.

Class	Seed terms
Event	{ dead, injured, missed, trapped, displaced, kill, victim, damaged, building, road, bridge, village, destroy, communication, home }
Action	{ donation, shelter, food, water, money, medical, volunteer, clothing, pray, heart, hope, sad, tears, crying, thoughts }

Parameter setting. In the experiments, the number of queries and number of data samples were set to 5 and 50, respectively using the HILDEX interface. Also, we assumed that the user's information needs are static and do not change during the experiment. Based on this assumption, we used pseudo-relevance feedback to select query and data samples. Accordingly, we assumed that the user selects the top 1 query (the 1st best query) among queries generated using query morphing and queries-as-answers operations, which ranked based on average precision and similarity score. We also assumed that the user utilizes the selected query to start the exploration process again until the value of the metrics improves. Once a query with high precision and a similarity score is found, we assumed that the user marks the top 10 retrieved tweets of that query results as examples for the query-by-example operation.

5.2.2 Evaluation and assessment

This use case was intended to profile crisis evolution through exploration queries. For evaluating the exploration results we used the following metrics used for information retrieval tasks: Mean Precision@K (MP@K), Mean Recall@K (MR@K), F-score@K and Mean Similarity@K[28].

Table 5.10 presents the performance details of HILDEX for synthetic queries with sizes one, two and three. For each synthetic query, we considered top-50 tweets retrieved by a generated query to calculate MP@50, MR@50, F-score@50 and MS@50.

Table 5.10: HILDEX performance based on mean precision@50(MP@50), mean recall@50(MR@50), F-score@50 and mean similarity@50(MS@50)

Class	# of given term in query	# of synthetic query	MP@50	MR@50	F-score@50	MS@50
Event	1	15	0.80	0.0099	0.0195	0.77
	2	105	0.82	0.0105	0.0207	0.78
	3	455	0.86	0.0115	0.0226	0.79
Action	1	15	0.95	0.0077	0.0152	0.78
	2	105	0.96	0.0078	0.0154	0.78
	3	455	**0.98**	0.0080	0.0158	**0.80**
		1150				

Action-queries performed considerably better than *event-queries* in terms of mean Precision@50 and Recall@50. However, as evidence from Table 5.11, the performance of *event-queries* in terms of the average mean Precision@50 using HILDEX improved compared with other query exploration techniques(query morphing, queries as answer and query by example) used as baseline (best MP@50 of 0.74 for event-queries in the query-by-example technique and best MP@50 of 0.82 for event-queries in HILDEX).

Table 5.11: A comparison between HILDEX performance against query morphing, queries as answers and query by example based on event and action class.

Class	Evaluation Metrics	Query Morphing	Queries as Answer	Query by Example	HILDEX
Event	MP@50	0.49	0.61	0.74	**0.82**
	MS@50	0.54	0.67	**0.80**	0.78
Action	MP@50	0.87	0.83	**0.96**	**0.96**
	MS@50	0.55	0.65	**0.82**	0.78

Also, note that morphing synthetic queries by adding terms increases their performance in terms of mean Precision@50 (see Table 5.10). Indeed, the best MP@50 is 0.98 for an action-query with three terms and best MP@50

[28]For example, mean precision at 10 documents (MP@10) corresponds to the average number of relevant results among the top 10 retrieved documents.

is 0.86 for an event-query with three terms. In addition, morphing both query classes by adding terms slightly increases the mean similarity@50.

5.2.3 Results and discussion

Figure 5.6 depicts a comparison between query morphing, queries-as-answers, query-by-example and HILDEX exploration workflow results. HILDEX results surpass the other exploration techniques in terms of mean Precision@50. Based on the presented results, we can conclude that each proposed exploration technique has some weaknesses; however, after combining them and generating HILDEX, they have the best performance together. It must be kept in mind that we used pseudo-relevance feedback to select the best query and suitable examples to retrieve results. We believe that if we collect explicit feedback from the user, HILDEX can perform better in all evaluation metrics.

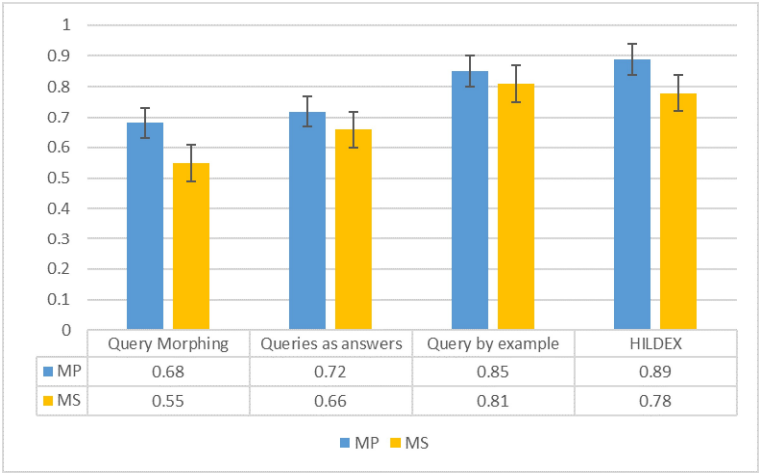

Figure 5.6: Comparison Mean Precision(MP) and Mean Similarity(MS) between all exploration techniques and HILDEX.

5.3 Understanding COVID-19 by exploring scientific data collections

Exploring an ever increasing amount of scientific papers that are produced about COVID-19 related topics is challenging for scientists. Indeed, the study presented in [118] shows that scientists spend a large part of the time (about 23%) in searching and reading research literature.

Use case objective. This use case aims at exploring scientific papers for understanding the characteristics of the disease by identifying topics and textual patterns and associating papers that address them. The objective is, for every query that represents a potential fact or hypothesis about the COVID-19, to find top-k papers that answer a query, and retrieve and ranking candidate sentences from papers which possibly answer the user query.

The exploration challenge is to extract, out of a large COVID-19 scientific literature corpus, the relevant papers possibly containing the answer to a user query about COVID-19. The semantic of queries in this use case was: *which are the top-k papers the are most probably close to a given topics?*

5.3.1 Exploring scientific articles

Overview of the use case exploration pipeline. When the user asks an initial query, HILDEX returns a set of papers (like in a traditional search engine) and returns sentences from the paper that are possible answers to the query. The user can review the sentences and decide on whether or not that paper is worth further reading. HILDEX first discovers several topics using LDA (Latent Dirichlet Allocation) and then uses them to find a set of related articles for a given query. After that, HILDEX uses this set to extract a collection of sentences containing the answer to the query. Then it uses sentence embedding and cosine similarity to select the most similar sentences. Finally, it selects the top K articles with the highest number of related sentences.

Identifying scientific articles topics. HILDEX first calculates the probability distributions of 50 topics over a given scientific literature corpus P_p. The probability P_p represents how well each topic describes a paper content. It also calculates the probability distribution of topics over the user query P_q. Then, it calculates the similarity score S to measure the similarity between two probability distributions (i.e., P_p and P_q representing the paper and query probability vectors, respectively). The similarity score is given as:

$$S = 1 - Jensen–Shannondistance \tag{5.1}$$

Jensen–Shannon distance is the square root of the Jensen-Shannon divergence, where Jensen–Shannon divergence measures the similarity between two probability distributions.

$$S = 1 - \sqrt{\frac{D(P_p \parallel m) + D(P_q \parallel m)))}{2}} \tag{5.2}$$

Where m represents the mean of P_p and P_q, D represents Kullback-Leibler divergence and S represent the similarity score between a user query and the research paper in the corpus. The range of S lies between 0 to 1. The value of the similarity score determines the similarity between the topic distribution of a paper and a query. Consequently, the probability of a paper represents its relevance concerning the query.

Using the similarity score, HILDEX selects the top relevant candidate papers to build a `related articles set`. This set contains papers that are the closest to the user query.

Human-in-the-loop. We worked with biologists and physicians from the Golestan University of Medical Sciences and developed an assessing method for data exploration algorithm. They tested our model on a collection of Kaggle tasks based on scientific questions developed with the World Health Organization and the National Academies of Sciences, Engineering, and Medicine to evaluate the accuracy and effectiveness of our documents exploration engine.

We experimented with various values of similarity scores ranging from 0.2 to 1. The domain experts from the biology and medical field manually reviewed the quality of extracted papers using different similarity scores. Finally, they agreed to considered articles having a similarity score of more than **0.5** on a scale of 0 to 1 as a candidate for being in the related articles set.

Candidate answers and relevant papers are the top-k papers possibly containing the answer to a user query.

- Candidate answers: HILDEX converts each relevant paper in the `top related papers set`, into a set of "representative" sentences. We create a list of tuples:

  `Top_relevant_paper_sentences: <sentence, paper ID>`

  Each sentence in the list is converted into a vector representation using our previously trained `word2vec` model.

- Relevant papers: Given the query vector representing the user query, HILDEX calculates a similarity score between the query vector and all the elements of the list *Top_relevant_paper_sentences*.

  Based on the similarity score, we choose sentences with the similarity score above a fixed threshold and then sort the sentences so that the

sentences with higher similarity scores are placed above in the list of candidate answers. Our experiments showed that sentences having a Cosine similarity score equal to or above 0.5 on a scale of 0 to 1, can be considered candidates to answer the query.

– Ranking relevant papers is based on the top candidate papers, to group the sentences with their respective research paper ($< paperID >$). Each paper has a set of answer candidates, ranked with respect to others based on the number of answer candidates. For example: If papers A and B have respectively ten versus seven-candidate sentences' answers, then A is ranked above B in the ranking of relevant papers list. The idea behind is that if a paper has a high number of candidate sentences, it means that its content is contextually more aligned with the user query. Consequently, the paper should be ranked higher in the relevant paper ranking.

5.3.2 Evaluation and assessment

For the case of the COVID related topics, there is not enough standardized question answering dataset. So, we worked with three field experts of medicine and biology from the National Institute of Genetic Engineering and Biotechnology, Tehran, Iran and Golestan University of Medical Sciences to assess HILDEX and the baseline exploration of search engines. We devoted scores related to each result according to their relevance to our queries to quantify our evaluation. Thus, three scores are assigned to our findings, according to the following formula:

$$Evaluation(article) = \left\{ \begin{array}{cc} 1 & Relevant \\ 0 & PartiallyRelevant \\ -0.5 & NotRelevant \end{array} \right\} \tag{5.3}$$

In the formula, the score of -0.5 is a penalty for completely irrelevant articles, 0 for papers that may be a candidate for parts of our findings (i.e., partially relevant) and score 1 concern the documents that are entirely related to the use case questions.

To have a standard set of diverse queries, we chose a subset consisting of 30 questions (see Table 5.13 and 5.15) out of 100 questions of Kaggle CORD-19 research challenge.[29] The challenge consists of 17 sub-tasks, out of which 9 sub-tasks (refer table 5.12) were related to information retrieval. Each of the 9 sub-tasks consists of 5-10 questions. We selected 3-4 questions for each of the 9 tasks. The selection pattern of our choice was according to two criteria. First, we chose topics with enough variety to cover all aspects

of the pandemic. Second, we selected questions that engaged scientific minds about the COVID-19. Since, out of 100 questions, various questions were overlapping and repetitive. Our biology and medical experts identified 30 most interesting questions that were most distinctive to overcome this issue.

Table 5.12: COVID-19 Tasks

Task Id	Task details
$Task1$	What is known about transmission, incubation, and environmental stability?
$Task2$	What do we know about COVID-19 risk factors?
$Task3$	What do we know about vaccines and therapeutics?
$Task4$	What do we know about virus genetics, origin, and evolution?
$Task5$	What has been published about medical care?
$Task6$	What do we know about non-pharmaceutical interventions?
$Task7$	What has been published about ethical and social science considerations?
$Task8$	What do we know about diagnostics and surveillance?
$Task9$	What has been published about information sharing and inter-sectoral collaboration?

To illustrate our scoring method, we show three results in response to a COVID-19 question. As given in the table 5.13, let us take *"the effect of seasons on the transmission of COVID-19"* as one of the searched query items. The following papers are three of our first five related papers extracted by the algorithm.

1. "Climate effect on COVID-19 spread rate: an online surveillance tool [120]."
2. "Projecting the transmission dynamics of SARS-CoV-2 through the post pandemic period [149]."
3. "Excess cases of influenza suggest an earlier start to the corona virus epidemic in Spain than official figures tell us: an analysis of primary care electronic medical records from over 6 million people from Catalonia [150]."

The first one gained score one because it has relevant data related to our

[29] https://www.kaggle.com/allen-institute-for-ai/CORD-19-research-challenge

Table 5.13: COVID-19 Questions

Id	Question
$Q1$	The incubation period of corona virus disease
Q_2	The effect of seasons on transmission of COVID-19
Q_3	Risk factors for severe disease and death
Q_4	Efforts to develop a SARS-CoV vaccine
Q_5	Risk-reduction strategies
Q_6	Misinformation relate to COVID-19
Q_7	The economic impact of COVID-19
Q_8	Use of diagnostics markers to detect early covid-19 disease
Q_9	Protocols for screening and testing of covid-19
Q_{10}	Outcomes data for COVID-19 after mechanical ventilation

query. The second article gained zero because it "seems" to be related to our questions, and the third one gained a -0.5 score as a penalty because of its irrelevant result. Indeed, it is not about our query.

5.3.3 Results and discussion

We have showcased HILDEX's capability of finding the most relevant candidate answer sentences out of selected papers (see Table 5.14 and 5.15). This shows that the quality of related output papers is not the only advantage of HILDEX. Our application response to question Q_1 in the table 5.13 is shown in figure 5.7. This application brings related papers and the most significant sentence of it.[30] The user can customise and refine the results by year of article publication and selecting only COVID-19 associated articles, which have terms like COVID-19, Coronavirus, SARS-CoV-2 and 2019-nCoV. The application can also be customised based on the number of relevant papers and sentences the user wants to see. Finding and reading related papers is one of the most critical but time-consuming scientific work processes. It is mainly about the researchers who work on systematic reviews and meta-analyses. Therefore, such a "helper" like HILDEX can facilitate research processes and drive the force of researchers toward scientific innovation rather than just wandering and being lost in an immense amount of scientific papers.

[30]In the time of writing this paper, we are working on its user-friendly-related features, so figure 5.7 is just an example of its functionality on Google Colab.

Figure 5.7: Screenshot of HILDEX when it builds on LDA and our ranking algorithm.

5.4 Exploring multilingual micro-texts

The COVID-19 pandemic has highlighted the extent to which the world's population is interconnected through the Internet and social media. Social media provides particularly fertile ground for the spread of information and misinformation [151]. It can even give direct access to content, which may intensify rumours and dubious information [152]. For people with non-medical experience, it is not easy to assess health information's authenticity. Seeking accurate and valid information is the biggest challenge with Internet health information during the pandemic [153].

Both misinformation[31] and disinformation[32], according to the Oxford English Dictionary, are false or misleading information. Misinformation refers to information that is accidentally false and spread without the intent to hurt, whereas disinformation refers to false information that is intentionally produced and shared to cause hurt [154]. Claims do not have to be entirely truthful or incorrect; they can contain a small amount of false or inaccurate information [155]. We use the general notion of misinformation and make no distinction between misinformation and disinformation as it is practically

[31] https://www.oed.com/view/Entry/119699?redirectedFrom=misinformation
[32] https://www.oed.com/view/Entry/54579?redirectedFrom=disinformation

difficult to determine one's intention computationally.

Use case objective. Multilingual tweet analysis and misinformation detection using HILDEX is intended to observe social media misinformation spread about the COVID-19 pandemic within communities with different languages. This use case calls less to human-in-the-loop actions, and it seeks to observe the trends of misinformation in a tweets data collection, implying classification and quantitative exploration.

5.4.1 Exploring misinformation through classification

HILDEX was calibrated with models adapted for dealing with COVID-19 scientific papers. For training the models the data collection was fragmented into training, validation and testing data collections in the ratio of 80%/10%10% respectively. The final count for the train, validation and test dataset was 7,602, 950, 950. We fine-tuned the Sequence Classifier from HuggingFace based on the parameters specified in [156]. Thus, we set a batch size of 32, learning rate 1e-4, with Adam Weight Decay as the optimizer. We ran the model for training for 10 epochs. Then, we save the model weights of the transformer, helpful for further training.

HILDEX Hyperparameters' Setting. Table 5.16 lists every hyperparameter for training and testing the models and thereby enhancing HILDEX. All the calculations and selection of hyperparameters were made based on tests and the model's best output. After performing several iterations on distinct sets of hyper-parameters based on the model's performance analysis, we adopted the one showing promising results on our dataset.

5.4.2 Evaluation and assessment

We experimented with the multilingual data with their respective linguistic-based BERT models. We set the model with the same training parameters as HILDEX enhanced with well-adapted models and pre-processed the data as stated previously. Each monolingual model was fine-tuned for 10 epochs with a batch size of 32, and it was applied to the classification dataset of their respective language. HILDEX results achieved an accuracy(%) of **82.17** (see figure 5.8) and F_1 score (%) of **82.54** on the test dataset. The precision and recall reported by the model were **82.07** and **82.30** respectively.

Table 5.17 shows the model's prediction over few examples from the test dataset along with their actual label. As shown in the table, in the entry numbers 1, 2, 3 and 4 our model could predict the correct label. However,

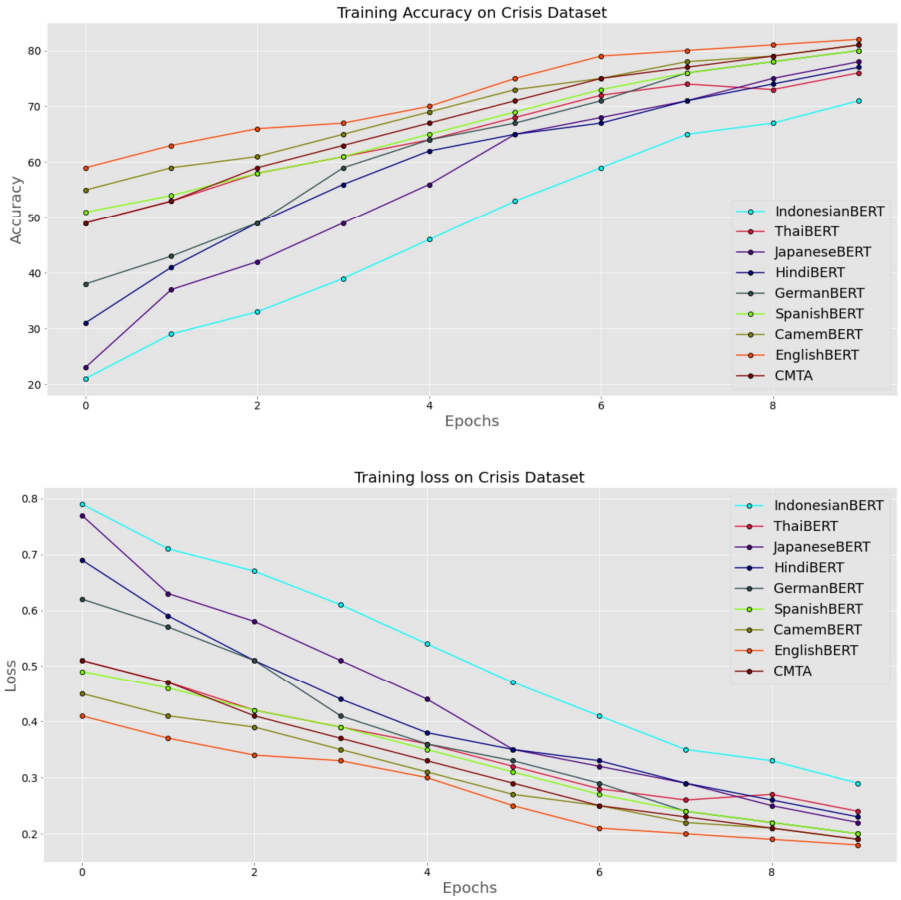

Figure 5.8: Training Accuracy(Upper) and Training loss(Lower)

in the case of entry number 5, the label predicted by our model was *False*, whereas the actual label is *Misleading*. If we would look at the misinformation at the entry number 5, which is a Spanish text *"El medicamento contra piojos sirve como tratamiento contra Covid-19"* and the corresponding English translation would be *"The anti-lice medicine serves as a treatment against Covid-19"*. This misinformation claims about a COVID-19 medicine, and since this could be *false* and *misleading* misinformation at the same time, our model predicted it as *false* misinformation rather than *misleading*.

Multilingual Misinformation Exploration We provide detailed exploration results of misinformation distribution across multilingual micro-texts (i.e., tweets). This analysis responds to the initial question: *how is misinformation about COVID-19 spread in communities speaking different languages.*

The exploration tasks performed using HILDEX were intended to do a quantitative exploration of the data collections to observe the distribution of COVID-19 misinformation across eight significant languages (i.e. English, Spanish, Indonesian, French, Japanese, Thai, Hindi and German) for five months in 2020 (i.e. February, March, April, May and June).

HILDEX was calibrated with a trained, multilingual model to predict and classify the misinformation types in tweets. Using HILDEX, we conducted sequential misinformation exploration tasks on the collection of over 2 million multilingual tweets (described in Section 5.1).

Figure 5.9 shows the month-wise distribution of misinformation types for each language. It showcases the overall (all 5 months together) spread of misinformation types across each language. We could see that German tweets have the highest number of *Misleading* tweets, whereas French have the least. Spanish tweets beat other language's tweets by becoming the largest source of *False* misinformation. Germany generated the least number of *False* tweets. Hindi tweets tend to have the highest number of *Partially False* tweets, whereas Thai have the least.

We could observe that for February, March and June months, our model predicted a large number of tweets as *False*, followed by *Misleading*, which is the second largest and the number of *Partially False* was the least (see Figure 5.10). Our model discovered that the number of *Partially False* tweets are more than *Misleading* tweets and *False* tweets were again in the majority for the tweets generated during April and May.

5.4.3 Results and discussion

Table 5.18 and 5.19 present the results provided by HILDEX giving a quantitative view of misinformation classes across all the languages, that can help to explore this data and understand the spread of false, misleading and partially false tweets about COVID-19 in different languages. The following specific observations were made concerning the languages:

- English: The misinformation distribution for English data indicates that there is a majority of **False** tweets during the five months, whereas the distribution of **Misleading** labelled data is slightly less than as compared to **False** labelled data. **Partially False** labelled tweets are moderately distributed, as, in April, we can see that there is a more significant number concerning other months.
- Spanish: According to the language wise-distribution shown in Figure 5.9, Spanish tweets have a greater frequency of **False** labelled tweets,

Figure 5.9: Month-wise Disinformation Distribution in Languages.

whereas the **Misleading** tweets and **Partially False** tweets shows the almost identical number of tweet across the five months.

– German: There was a surge of **Misleading** labelled tweets during February, and the count remained the same throughout the five months. There was also an increase in **Partially False** tweets in March, but it decreased in successive months, leading to minor **False** labelled tweets.

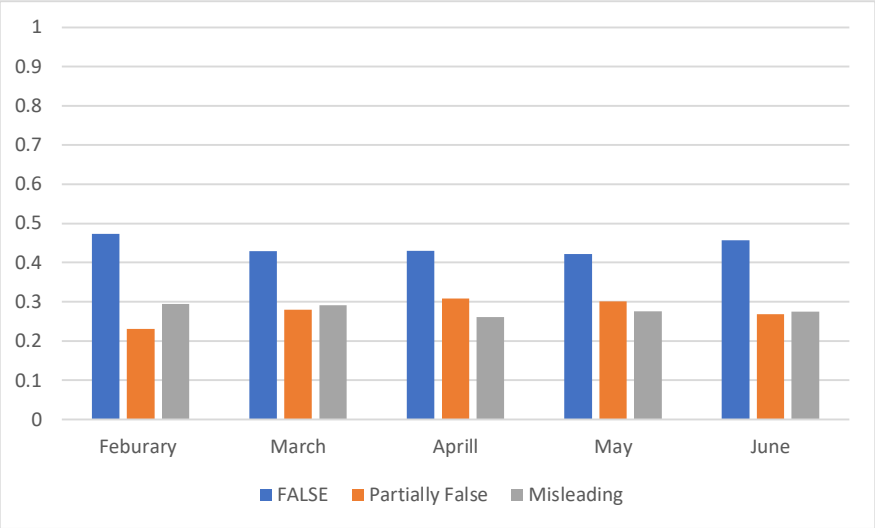

Figure 5.10: Month-wise Disinformation Distribution.

- Japanese: According to the language wise-distribution shown in Figure 5.9, on average throughout the five months, approx 20% of Japanese tweets are labelled **False**. Similarly, approx 30% of the Japanese tweets are labelled **Partially False**, leading to the majority of 50% data are labelled as **Misleading**. We can also see a considerable increase in **Misleading** tweets in March, tweeted in the Japanese language.
- Indonesian: According to the distribution of Indonesian tweets, approximately 10% of tweets are labelled as **Misleading**, and on the contrary, there is a large distribution of **False** labelled tweets. Approximately 34% of the Indonesian dialect data is labelled as **Partially False** throughout the five months.
- French: Figure 5.9 shows the misinformation distribution across all five months in the French tweets. The largest majority of the tweets were classified as **False** misinformation. Among **Partially false** and **Misleading**, the least number of tweets were labelled as **Misleading**.
- Hindi: The frequency of Hindi tweets is low in the dataset used in our experiment. However, our model can predict or label Hindi tweets. Tweets in Hindi have low numbers of **Misleading** tweets, whereas the **Partially False** tweets class has a great frequency. **False** labelled tweets are slightly low compared to **Partially False** tweets in this dialect.

– Thai: The distribution of Thai tweets shows that our model prediction is majorly oriented towards the **Misleading** tweets. The distribution of **Misleading** labelled tweets is the greatest among the labelled classes, in contrast to **Partially False** tweets. **False** labelled tweets are comparatively moderate in this language.

5.5 Conclusions

This chapter described the validation of the HILDEX exploration workflow that we conducted through use cases. We prepared textual data collections regarding crisis management and COVID-19 scientific articles and misinformation micro-texts for assessing the framework. The objective was to validate to which extent HILDEX explores textual data collections through queries - sample results to respond to users questions.

According to the characteristics of the data collections (i.e., micro-texts, multilingual, guided by topics), we enhanced HILDEX with text processing models adjusted to the knowledge domain of the data collections. We were motivated by the fact that there are text processing models targeting crisis (word2vec for crisis data), textual scientific papers and multilingual texts (e.g., MBERT). We produced labelled data collections ready to be explored.

Our validation use cases and experiments revealed that the workflow of HILDEX that combines different data exploration techniques could perform very well and generates high-quality queries and answers (e.g., crisis and COVID-19 interactive question-answers use cases). These assessment results relied on scores and metrics and the opinion of real users. These cases showed the importance of the role of the human in the loop facility of the exploration workflow. Finally, we also observed that exploration requirements could vary from those where interaction and the combination of query exploration algorithms are relevant (e.g., crisis management and scientific papers exploration). Other exploration requirements are more static and expect an explanatory report on a concrete question (e.g., how does misinformation spread in different languages?).

Table 5.14: Related sentences extracted for the questions in the table 5.13

Id	HILDEX Answer
Q_1	In our analysis of 44 patients with clear contact history, we found that the mean incubation period of COVID-19 was 8 [119].
Q_2	Our findings of decreased replication and spread rates of COVID-19 in warm climates may suggest that the inevitable seasonal variance will alter the dynamic of the disease spread in both hemispheres in the coming months [120].
Q_3	Evidence from China, Italy and the USA indicates that older individuals, males and those with underlying conditions, such as CVD, diabetes and CRD, are at greater risk of severe COVID-19 illness and death [121].
Q_4	Here, we discuss therapeutic and prophylactic interventions for SARS-CoV-2 with a focus on vaccine development and its challenges [122].
Q_5	In addition, we provide suggestions to aid further development of epidemic prevention and control strategies, and scientific decision-making [123].
Q_6	The World Health Organization have emphasized that misinformation - spreading rapidly through social media - poses a serious threat to the COVID-19 response [124].
Q_7	We identify a total of 35 potential determinants that describe a diverse ensemble of social and economic factors,including healthcare infrastructure, societal characteristics,economic performance, demographic structure etc [125].
Q_8	Those findings indicate that our N antigen assay is an accurate, rapid, early and simple diagnosis method of COVID-19 [126].
Q_9	We established routines for SARS-CoV-2 RNA extraction-free single-reaction RT-qPCR testing [127].
Q_{10}	All patients demonstrated stable physiology and ventilation for the duration of shared ventilation [128].

Table 5.15: Answers for COVID-19 related questions

Question	Exploration Answer
11. Persistence of virus on surfaces of different materials	A report by van Doremalen and colleagues found survival of both SARS-CoV and SARS-CoV-2 of up to 2 days (on surfaces) and 3 days (in aerosols generated in the laboratory), but again with a large inoculum [129].
12. Role of the environment in transmission	The Respiratory diseases are often simply assumed to be transmitted via "close contact"; however, the complex transmission mechanisms often involve with more than one transmission route including direct or indirect contact, large droplet, and airborne routes [130].
13. Approaches to evaluate risk for enhanced disease after vaccination	NHPs could be utilized to evaluate COVID-19 vaccine candidates without adjuvants and guide in the selection of vaccines that elicit desired attributes that could reduce the risk of vaccine-mediated enhanced disease [131].
14. Effectiveness of drugs being developed and tried to treat COVID-19 patients	Remdesivir (GS-5734™) is an antiviral drug developed by Gilead Sciences initially developed to treat Ebola, but experimental tests are also being carried out to treat diseases such as MERS and COVID-19 [132].
15. Outcomes data for COVID-19 after mechanical ventilation adjusted for age	As age increased, so did the proportion of patients who required ICU admission, invasive mechanical ventilation, and vasopressors. Median age was 76 years (IQR, 66-85); 58% (n=244) were male; 71% (n=299) were admitted to the ICU; and 59% (n=246) received invasive mechanical ventilation [133].
16. Guidance on the simple things people can do at home to take care of sick people and manage disease.	The best health advice for older, frail patients was to stay home and health care providers offered televisits and telephonic symptom management to avoid unnecessary emergency department visits [134].
17. Policies and protocols for screening and testing.	In this study we propose a novel group testing protocol using a commercially available RT-dPCR assay and compare empirically the sensitivity of individual identification through RT-PCR with group testing by RT-dPCR for three groups sizes of 8, 16 and 32 samples [135].
18. Efforts to track the evolution of the virus	Mutation and adaptation have driven the co-evolution of coronaviruses (CoVs) and their hosts, including human beings, for thousands of years [136].
19. Effectiveness of case quarantine of exposed individuals	If 90% of cases are asymptomatic or undetected, as could happen in a location making no effort to follow people during their isolation, the efficacy of quarantine would be about 70% [137].
20. Effectiveness of inter/inner travel restriction	During the peak of the COVID-19 outbreak in Europe, about 3-6% of air passengers were SARS-CoV-2 positive on repatriation flights [138].
21. Effectiveness of school distancing	We fit our model with the data until August 29th, and then simulate what would happen in the event that schools open in mid-September [139].
22. How does temperature and humidity affect the transmission of 2019-nCoV?	Analyzed meteorological data of 30 cities in China and suggested that low temperature, mild diurnal temperatures, and low humidity likely aid the transmission of novel coronavirus disease 2019 (COVID-19) [140].
23. What is the efficacy of novel therapeutics being tested currently?	Remdesivir and favipiravir are the most promising antiviral drugs that have been tested in clinical trials so far [141].
24. Risk factor studies related to impact of diabetes	Conclusion: Older people above 65 years old and diabetic patients are significant risk factors for COVID-19 [142].
25. Risk factor studies related to impact of male gender	The multivariate analyses, age over 50 years, male gender and low-medium socioeconomic status were also positively associated with the risk of COVID-19 infection [143].
26. Risk factor studies related to impact of kidney disease	Kaplan-Meier analysis demonstrated that patients with kidney disease had a significantly higher risk for in-hospital death [144].
27. Risk factor studies related to impact of cancer	Combined with previously published results , we can conclude that patients with cancer have an increased risk of COVID-19 [145].
28. Risk factor studies related to impact of overweight	Our findings are consistent with other reports from New York that did not find obesity to be an independent risk factor for mortality, though reports from reviews suggest obesity does play a role in mortality [146].
29. What do we know about viral shedding in stool?	In addition, we identified six studies presenting indirect evidence on the potential for SARS-CoV-2 transmission by children, three of which found prolonged virus shedding in stools [147].
30. What is the longest duration of viral shedding?	This happens to coincide with the fact that the viral RNA shedding of children is much longer than that of adults(13, 14) [148].

Table 5.16: Hyper-parameters for training

Parameters	**Value**
Pool Size of Average Pooling	8
Pool Size of Max Pooling	8
Dropout Probability	0.36
Number of Dense layers	4
Text Length	128
Batch Size	32
Epochs	10
Optimizer	Adam
Learning Rate	1×10^{-4}

Table 5.17: Misinformation data examples along with model's prediction and actual label

Test Data	**Actual Label**	**Prediction**	**Accuracy(✓/✗)**
Dr. Megha Vyas from India died due to COVID-19 while treating COVID patients.	False	False	✓
El plátano bloquea "la entrada celular del COVID-19"	False	False	✓
Asymptomatic people are very rarely contagious, said the WHO.	Partially False	Partially False	✓
Patanjali Coronil drops can help cure coronavirus.	Misleading	Misleading	✓
El medicamento contra piojos sirve como tratamiento contra Covid-19.	Misleading	False	✗

Table 5.18: Language-wise predicted misinformation labels of tweets in February, March and April.

Lingo	**February**			**March**			**April**		
	Misinformation			**Misinformation**			**Misinformation**		
	False	**Partially False**	**Misleading**	**False**	**Partially False**	**Misleading**	**False**	**Partially False**	**Misleading**
Spanish	58346	6653	13740	67956	10913	8826	34125	5437	3604
German	517	581	2505	862	1438	3043	584	892	2664
Japanese	1920	3079	5245	448	692	2650	1635	2850	5840
Indonesian	11157	3226	1951	12573	4336	1582	9073	3367	1273
English	88369	62747	76640	92428	96571	105143	77368	74947	63473
French	4464	3472	1155	12024	10270	1670	6650	5300	763
Hindi	500	870	202	756	909	348	2211	2868	705
Thai	1950	1074	2780	6036	736	7678	2263	554	2917

Table 5.19: Language-wise predicted misinformation labels of tweets in May and June.

Lingo	**May**			**June**		
	Misinformation			**Misinformation**		
	False	**Partially False**	**Misleading**	**False**	**Partially False**	**Misleading**
Spanish	57821	8214	7107	54965	8828	6759
German	1076	1426	4430	616	657	2028
Japanese	8984	12324	18125	1741	2496	3389
Indonesian	12695	4574	1805	9114	3038	1000
English	140494	128326	119391	135172	101896	109483
French	8475	7667	842	4952	3535	483
Hindi	4560	6057	1343	2501	2739	751
Thai	2825	470	1830	2103	486	3122

Chapter 6

Conclusion and Future Work

The initial objective of this work was to propose a data exploration approach gathering different querying techniques like queries-as-answers, query morphing and query-by-example for assisting data exploration.

Accordingly, we proposed a human in the loop exploration workflow that coordinates a series of textual data exploration algorithms implementing different query rewriting techniques. An initial keyword query triggers this workflow. It applies techniques to analyse data collections and generates sets of queries that can produce better precision and recall scores. The user adjusts partial results, and at the end, the user obtains a set of queries and documents that fulfil her requirements. The exploration workflow is implemented by an associated framework named HILDEX.

6.1 Summary

6.1.1 Data exploration algorithms

We proposed a series of algorithms that propose query rewriting techniques: query morphing, query-as-answers and query-by-example. Algorithms rewrite queries adapted to better explore textual data collections according to initial queries provided by a user. Each algorithm was experimentally tested using a dedicated use case and assessed using scores like precision and similarity. In order to experiment on these data exploration algorithms, we prepared textual data collections regarding crisis management and COVID-19 information and misinformation. We used state of the art machine learning techniques

for this purpose and compared the interest of applying different techniques for processing this type of data.

6.1.2 Human-in-the-loop based textual data exploration workflow

We then proposed a human-in-the-loop based data exploration workflow that integrates into a pipeline the proposed algorithms. The workflow relies on a pre-processed data collection where its content is represented as a model (e.g., word2vec, Bert), and a vocabulary that organises the frequent and representative terms of its content.

The workflow consists of exploration phases devoted to rewriting an initial query into a morphed/extended version or on a set of alternative queries that use similar, general and equivalent terms that increase the scope of results. The user analyses the proposed queries and associated result samples and chooses those queries that seem provide results close to her expectations.

The workflow encourages the user's participation by providing queries and samples of the possible results obtained to help them calibrate these queries and retrieve data with good similarity, precision, recall and F1-score. Scores and explanatory feedback are included in exploration results to help the user choose and adjust her queries. Once the user has chosen the queries that provide acceptable results, the workflow includes phases for retrieving content using query-by-example techniques.

6.1.3 HILDEX data exploration framework

Algorithms and workflow are wrapped together by the framework HILDEX. We designed and implemented experiments for assessing the framework. The objective was to assess to which extent HILDEX proposes relevant queries (e.g., sample results to respond to users questions). The experiments were done with pseudo-relevance feedback and scientists of the National Institute of Genetic Engineering and Biotechnology, Tehran, Iran and Golestan University of Medical Sciences. Scientists provided feedback about exploration operations through questionnaires that were processed for obtaining satisfaction metrics.

6.2 Future work

In this section, we discuss the future works that might be considered for this thesis. Future work will tackle query expressions that will lead to retrieval

of textual documents and their analysis for easing the specification of data science queries that can be expressed on top of data collections. For the time being, HILDEX can ask questions of the type *which queries can be used for retrieving the right information (and as much as possible) a user is looking for?* The next step is to assist users to answer exploration questions like, *which are the analytics (modelling, clustering, prediction) questions that can be asked on a data collection?*.

We are currently using HILDEX to explore textual collections in the context of medical, geosciences and digital humanities data collections in the context of the projects ADAGEO[1], DOING[2] and LIFRANUM[3].

This work and, the state of the art, has shown the expectations of a tremendous productions of data announced due to the evolution of technology in the last years. This data collection production calls for efficient, agile, and accurate ways to explore their content and exploit them to extract knowledge. However, from our perspective, the real challenge is not introduced by the volume and the velocity of data collections, nor by its variety but about the conditions in which such data collections will be explored and exploited.

6.2.1 Extending HILDEX

The current version of HILDEX can be extended to provide more options for exploring data collections. Two main features can be added for enhancing it related to the type of data collection that can be explored and the choice of exploration operations to apply along the exploration process according to the content of the data collection, its type, and the user's objectives.

Exploring multimedia data collections. HILDEX currently manages textual data collections using adapted natural language processing models to pre-process them and prepare them for exploration through keyword queries. A first significant extension can be to adapt the data preparation phase with models adapted to images, videos, and other documents. Machine learning models and neural networks can be applied to process multimedia data. HILDEX could provide a module inspired by existing data labs where collections are uploaded and pre-processed, completed with quantitative and qualitative meta-data extracted automatically or manually by data providers.

[1] https://adageo.github.io
[2] https://www.univ-orleans.fr/lifo/evenements/doing/
[3] https://projet-lifranum.univ-lyon3.fr/

Recommending data exploration operations. The current exploration workflow implemented by HILDEX uses scores and feedback of partial results to decide which exploration branch to propose. Indeed, it either uses query-by-example technique when the user's initial query results have good scores and the percentage of relevant documents within retrieved results is acceptable. The "worse" case proposes a set of query rewriting strategies to propose queries that can provide better results. Instead of this pre-defined pipeline, the idea would be to consider the user queries, data collection and retrieval scores and even other exploration workflows applied with similar queries to recommend possible exploration operations and let the user define her data exploration workflow.

6.2.2 Exploring data, experiments and models for deriving the right queries

The availability of data collections and associated data regarding the experiment and models applied on top of them are assets that can enhance exploration tasks. The exploration challenge is to go beyond determining the kind of classic keyword, relational, and aggregation queries that they can answer. A more ambitious challenge is determining to which extent data collections can answer series of predictive, classification, correlation questions and how much preparation (data engineering) they need to participate in such tasks.

Deriving data science queries. Today, we see an increasing implementation of data science queries and associated tools to automate the analysis of data, and inferring or predicting new information. Too much data, regardless of how it is structured, creates storage problems and presents challenges when data consumers want to extract knowledge. Intelligent data filtering involves better distinguishing between useful data and data that can be filtered. The usefulness of data is, in general, related to users needs and objectives.

Therefore, starting a decision support project with general questions representing the knowledge that domain experts intend to obtain from the stored data seems promising. The domain experts' questions represent requirements that can guide (1) the design and integration of databases containing data to be exploited and (2) the pipelines to be applied to analyse and exploit the data to answer these questions.

New exploration techniques must be proposed to help choose and build data collections that can be used for applying specific models, for example, mathematical properties among attributes (e.g., linear independence of

attributes vs target attribute for regression). They can also help to derive data science pipelines interactively according to the content of given data and complementary information of experiments and models applied to the data in other experiments.

Towards a data exploration template language. Expressing data exploration requirements is not a straightforward task; this expression results from an interactive process where data are analysed and samples are proposed to help users adjust requirements. The final result is a data collection and the potential queries that can be asked on top of it.

We are willing to establish an iterative process, based on human-in-the-loop. This loop goes through the design of forms in successive stages of refinement, allowing specialists to refine their questions and keywords. It will then be possible to classify the questions (i.e. aggregation, exploration, modelling, prediction) and rewrite them into possible data exploration queries.

The primary goal is to contribute to the construction of user-friendly tools with declarative query languages where the user only needs to specify the high-level task, leaving the translation into pipelines to an exploration system.

APPENDIX A

Appendix

A.1 Systematic review: Data exploration

The steps of the Systematic Mapping (SM) process introduced in [157] are the definition of research questions, conduction of search, screening of papers, the definition of a classification scheme and data extraction. Figure A.1 presents all steps of the process performed in this work.

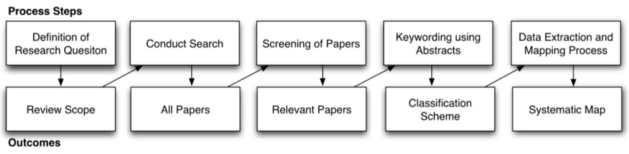

Figure A.1: Systematic mapping process.

The results of our SM study are presented as bubble charts in the next section. The visual representation using bubble charts has two main benefits. First, it easily shows the way the relevance of mapping categories has changed over time. Second, it reveals research opportunities hidden on the blanks found among classification categories, pointing topic areas and types of publications yet to be investigated.

A.1.1 Definition of research questions

The goal of our systematic mapping study (SM) is to identify different human guided data exploration techniques. In order to achieve this objective we stated following research questions:

RQ1: What are the current approaches for human guided data collections exploration?

RQ2: Which types of data queries can be done for exploring data collections? (keyword search, relational, statistical, descriptive (model), predictive?

RQ3: How have user requirements been considered in the evaluation of queries/data exploration? Which type of process is done by a human?

RQ4: How is the number of publications distributed over the years?

RQ5: Which type of urban data set has been used? (geo-temporal, traffic,...)

A.1.2 Conduction of search

The study led to the proposal of a classification of techniques and a quantitative (statistical) study of the literature that was used to answer the research questions. For applying the second phase of the systematic mapping methodology that consists in defining search queries (conjunctive and disjunctive) to be evaluated on top of publications search engines, we implemented a semi-automatic strategy. The strategy was intended to define the pertinent terms to be used to express the queries. It consisted in:

- Looking for terms of the domain defined by ACM and IEEE taxonomies and key words used by ISI Thompson/Web of Science quartile 1 (Q1) journals and conferences. We selected 12 terms common to these taxonomies and key words.
- Implementing a program that combines exhaustively the terms into 1144 queries.
- Implementing a program that uses search engines API's to execute batches of queries.
- Prune the results with too few and too much answers (379 papers as shown in Table A.1) and fusion results.

Based on the scope, the research string used in our study was expressed as the following conjunctive and disjunctive query:

" *("data exploration"* **OR** *"data analysis"* **OR** *"Query processing"* **OR** *"Information retrieval"* **OR** *Search)* **AND** *("Human in the loop"* **OR** *"Human-Computer Interaction"* **OR** *interactive)* **AND** *("urban computing"* **OR** *smart city* **OR** *urban* **OR** *"Emergency management")*"

We submitted this search string in four scientific databases: ACM Digital Library, IEEE Xplore Digital Library, Science Direct and Scopus. The search was limited to the title, abstract and keywords, which provide enough information to indicate whether a paper is in the context of the work or not. Also, we limited our search to papers published between years 1980 and 2019. As a result, we have obtained a total of 379 papers. The distribution of papers retrieved from each source is shown in the column "Retrieved" of Table A.1. In this total are excluded duplicate's papers that appear in more than one scientific database.

Table A.1: Sources and number of publications.

Source	*Retrieved*
ACM	141
IEEE	161
Science Direct	18
Scopus	59
Total	**379**

A database of 379 papers was obtained and processed using Rayyan [158] for the next phases of the SM method. Each relevant paper was analyzed to extract its information and contribution.

A.1.3 Screening of papers

The relevant papers were selected in two distinct phases. First, we analyzed the works just looking for information in the title and abstract. In this step, our criterion was to exclude papers that do not deal with data exploration. As a result, we have excluded 271 papers from the study, leaving 108 papers for the next phase. In the second phase, we analyzed the papers looking for information beyond the title and abstract. We applied inclusion and exclusion criteria to select the relevant papers for our study. These criteria are shown in Table 4. As a result of this phase, we have excluded 57 more papers from the 108 selected in the first phase. Finally, a total of 51 publications were selected for this study. Figure A.2 shows the number of publications selected from each step.

Table A.2: Inclusion and exclusion criteria

Inclusion criteria
- Text in English;
- Peer reviewed journals, conferences or workshops;
- Focus on data exploration OR Human in the loop;
Exclusion criteria
- Abstracts, tutorials, short papers and thesis;
- Technical reports and literature reviews;
- Papers that do not deal with data exploration OR Human in the loop;
- Chapter of book will not be studied, only articles are considered for review;

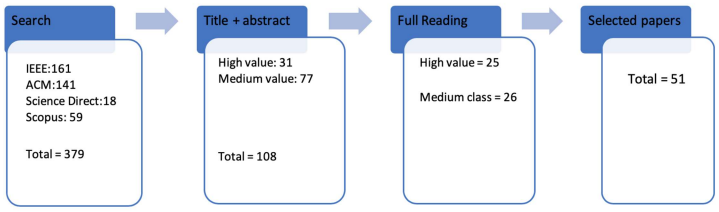

Figure A.2: Screening steps and number of publications selected from each step.

A.1.4 Classification scheme and data extraction

Based on our research questions and our previous knowledge in data exploration and analysis, we have proposed several categories to form an initial version of our classification scheme. This classification scheme was refined by reading the abstracts of the papers in order to find keywords and concepts that better reflect the contribution of the paper, in a process similar to the keywording presented in [157], where the keywords and concepts obtained from all papers are combined to build a classification scheme that reflects a knowledge base of the contribution of the research. Finally, we analyzed each relevant paper to extract its information and contribution. In this way, we have sorted the papers into the categories (called facets) presented in our classification scheme. This process was done with the help of Rayyan [158], a free web application to help systematic review authors, putting each facet and its dimensions as a label in each paper in rows. The results of this process are summarized in Tables A.3, A.4, A.5, A.6 and A.7.

Papers were sorted into categories (called facets) of classification scheme that I proposed. The proposed classification scheme consists in 5 facets with dimensions that describe our analytic vision of existing work. We defined a dummy facet that represents the type of contribution proposed by papers:

Table A.3: Facet: main contribution.

Dimension	References
Approach/Method	[159–167], [168–176], [177–184], [185–188]
Tool	[189–193]
Framework	[172, 194–201], [202–206]
Process	[207, 208]

The next facets are intended to classify the type of queries addressed by existing work (F2), the type of algorithms used in the case of analytics queries used for exploring data collections (F3), the knowledge domain of data collections and data types (F4), and the exploration processes done with human intervention (F5).

Table A.4: Facet: type of process is done by a human.

Dimension	References
Giving Feedback	[160, 177, 179, 181, 195, 199, 201, 205–207]
Visual exploration	[159, 161–165, 168, 171, 174–176, 182, 183, 187, 189–194, 198, 200, 202, 203, 208]
Specification of parameters	[166, 171–173, 178, 180, 184, 187, 193, 196]
None/Not Specified	[169, 170, 196]

Table A.5: Facet: types of data queries.

Dimension	References
Keyword search	[160, 162, 163, 165–167, 176, 187, 193, 196, 200, 207]
Relational	[163, 165, 168, 180, 191]
Descriptive	[160, 163, 164, 168, 199]
Statistical	[161, 165, 168, 178, 187, 200]
Predictive	[168]
Geo/spatio - temporal	[159–161, 163, 164, 167, 171, 176, 177, 182, 183, 187–189, 192–194, 198, 202]
Visual query	[163, 179, 197]

Table A.6: Facet: Data set type.

Dimension	References
Newspaper	[187]
Disaster	[159, 175, 181, 187, 189, 205]
Human mobility	[171, 191]
Twitter	[174, 177, 184]
Phone call	[169, 184, 185, 192]
Geo/Spatio Temporal	[159, 160, 164, 167, 169, 183–185, 189, 190, 193, 194, 198–201, 206]
Transport data	[161, 162, 169, 171–173, 177, 178, 182, 188, 193, 202, 206, 207]
Social network	[166, 195, 202, 203]
City infrstructure	[168, 204]
Image/Video	[163, 164, 176, 179, 203, 208]
Other	[170, 194, 196]

Table A.7: Facet: Algorithm types.

Dimension	References
Graph-based	[166, 173, 183, 184, 198, 202]
Machine Learning	[162, 165, 167–172, 176, 179, 184, 187, 190–193, 200, 205, 207, 208]
Heuristic	[199, 202]
Visualization	[159–162, 169, 173–178, 180–195, 201, 203, 204, 206]
Approximation	[196]

A.1.5 Analysis of the results

This section analyzes the facets quantitatively, presenting charts about individual facets and correlations between facets. Most of the individual facets are presented in bar charts and pie charts. Correlations between facets are presented as bubble charts, showing the intersection of the dimensions for two different facets. We assume that one paper can be counted in zero or more dimensions of one facet; this is the reason why the number of occurrences in a correlation may be different from the number of papers for each dimension. We combined facets to answer our research questions and to reveal interactions and relationships among the different analysis perspectives. For the sake of clarity, our investigation explores a pair of facets at a time. During the analysis, the data shown assume that the same paper can be classified in more than one dimension.

A.1.5.1 Quantitative analysis

The chart in Figure A.3 shows the distribution of publications over the years. We observe that distribution fluctuated wildly, however, the trend is upward. Also we can see an apparent drop of papers in 2019 since it may be partly

due to the fact that some papers were not available online at the time the search was performed.

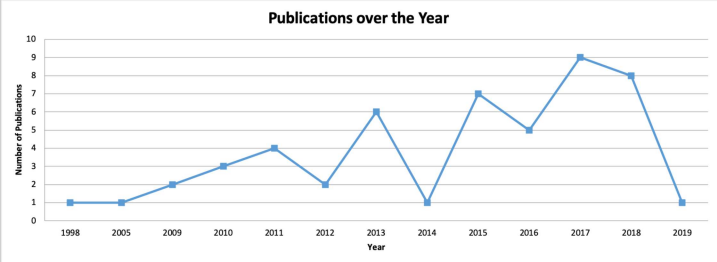

Figure A.3: Publications over the years.

Figure A.4 presents the distribution of the main contribution of the papers. Most of papers (60%) present some approach or method as its main contributions. Some papers also present frameworks (26%), tools (10%) or process as other main contribution.

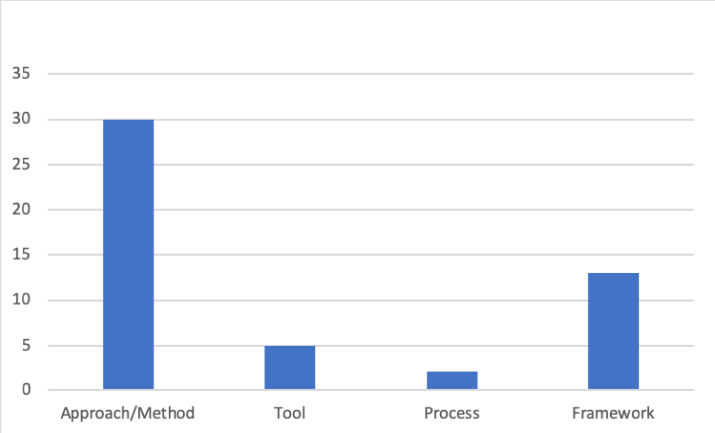

Figure A.4: Distribution of the main contribution.

The chart in Figure A.5 shows the distribution between different type of algorithms. This figure reveals that most of investigated papers use visualization (51%) and machine learning (34%). On the contrary, only a small number of papers have proposals graph-based(10%), heuristic(3%) and approximation(2%) algorithms.

The distribution between the data set types is presented in Figure A.6. Geo/Spatio temporal is the most frequent occurrence (28%), followed by transportation data (22%). Less than 3% of the papers are related to newspaper dataset, which shows a lack of interest in this type of data set.

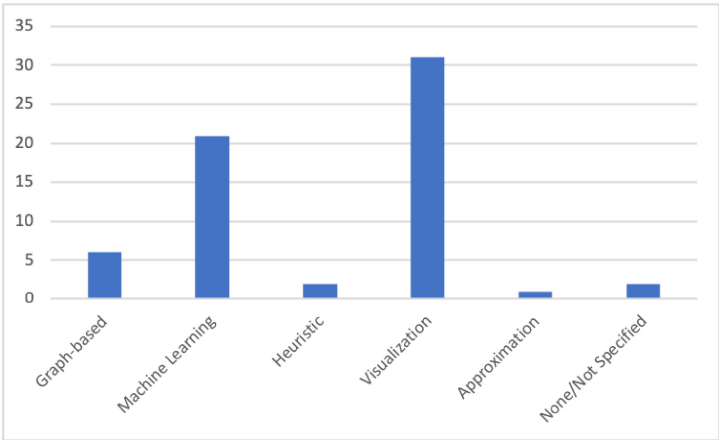

Figure A.5: Distribution of the algorithm types.

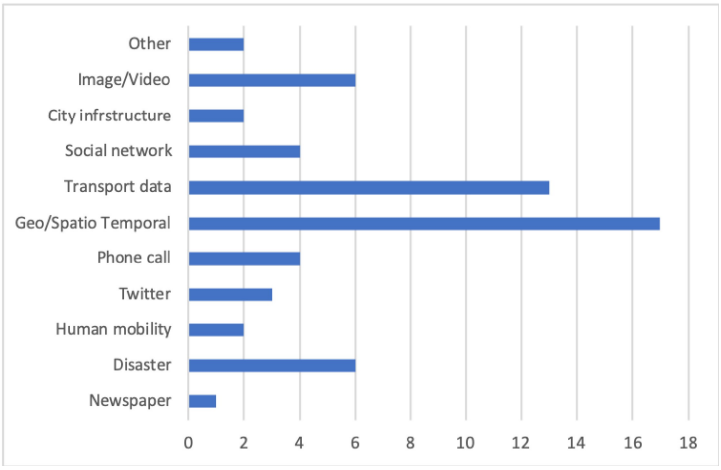

Figure A.6: Distribution of data set types.

Figure A.7 shows the different types of data queries used by the papers in our study. Geo/spatio - temporal are the most applied queries (37%), followed by keyword search (23%) and statistical (11%) queries. Less than 2% of the papers are related to predictive queries, which shows a lack of interest in this type of queries.

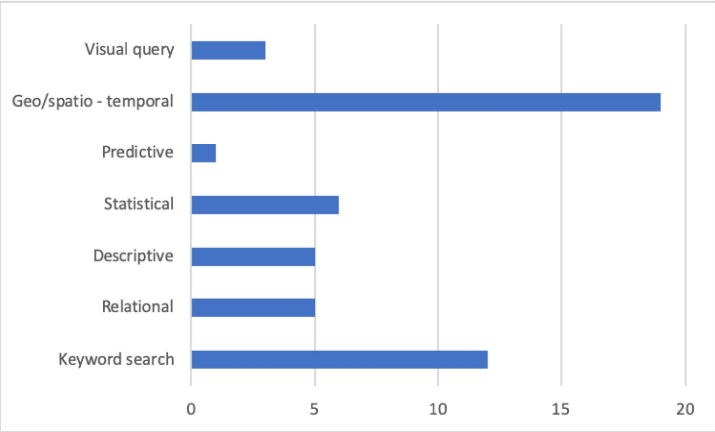

Figure A.7: Distribution of types of data queries.

Figure A.8 illustrates the distribution between types of process is done by a human. The majority (52%) of the papers use visual exploration, followed by specification of parameters (21%) and giving feedback (21%). Some of the works (6%) did not specify any specification type.

A.1.5.2 Analysis by combining facets

In this section, the proposed facets are combined to reveal interactions and relationships among the different analysis perspectives that we propose for studying human guided data exploration. For the sake of clarity, our investigation explores a pair of facets at a time. During the analysis, the data shown assume that the same paper can be classified in more than one dimension.

Type of process is done by a human and algorithm types A correlation between type of process is done by a human and algorithm types is presented in Figure A.9. It is possible to see a predominance of visual exploration with visualization and machine learning algorithms. Also, we can see that machine learning and visualization are very popular for exploring data. Also, we can observe that few proposals consider human in the loop (giving feedback) for guiding data exploration.

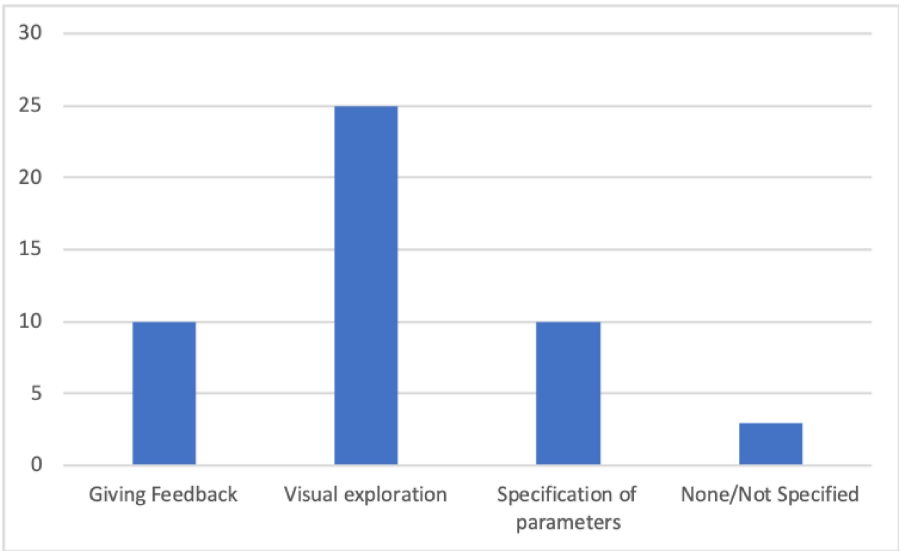

Figure A.8: Distribution of type of process is done by a human.

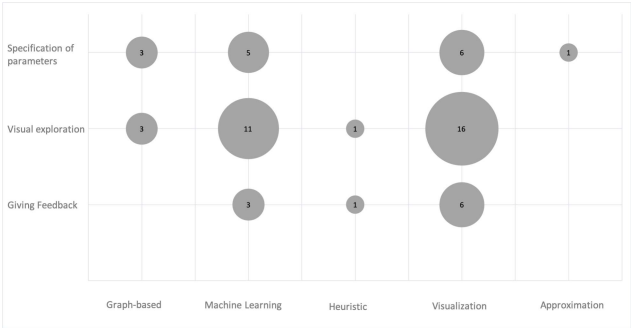

Figure A.9: Correlation between type of process is done by a human and algorithm types.

A.1.5.3 Type of process is done by a human and of data queries types

The combination of algorithm types with types of data queries facets is depicted in Figure A.10. It is possible to see that machine learning is correlated with all type of queries, followed by visualization, which just is not correlated with one.

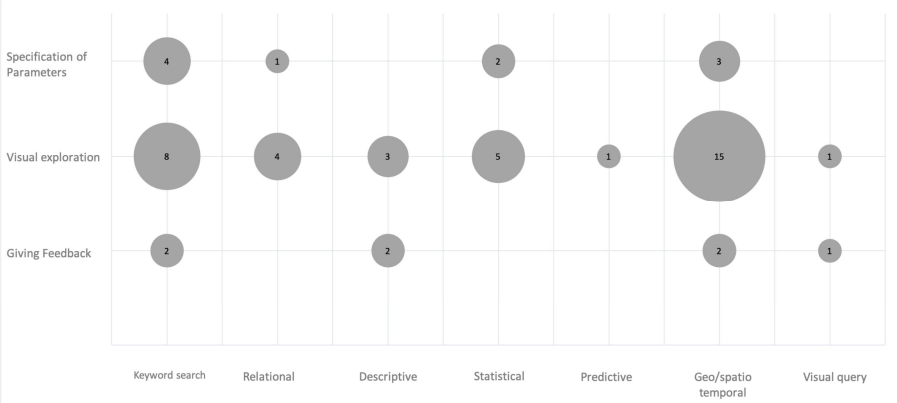

Figure A.10: Correlation between algorithm types with types of data queries.

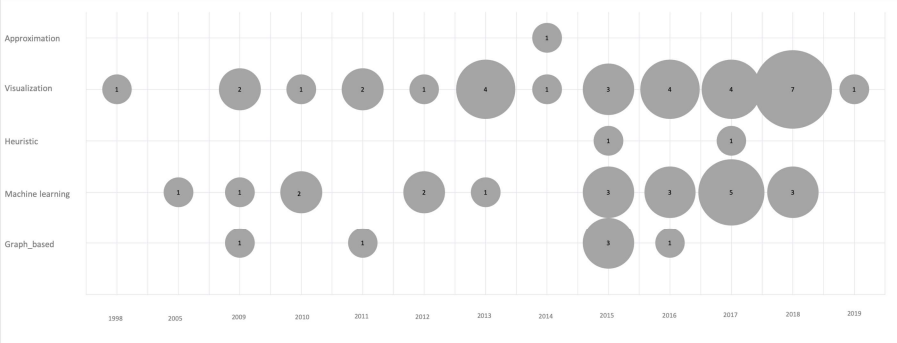

Figure A.11: Algorithm types per year.

A.1.6 Conclusions

The SM results presented in this work can be the starting point to motivate new studies, support the investigation of specific problems not sufficiently explored yet. Our quantitative analysis helps to see the global distribution of the research in the area, as well as the main conferences and journals that publish the results of that research. The qualitative analysis gives some

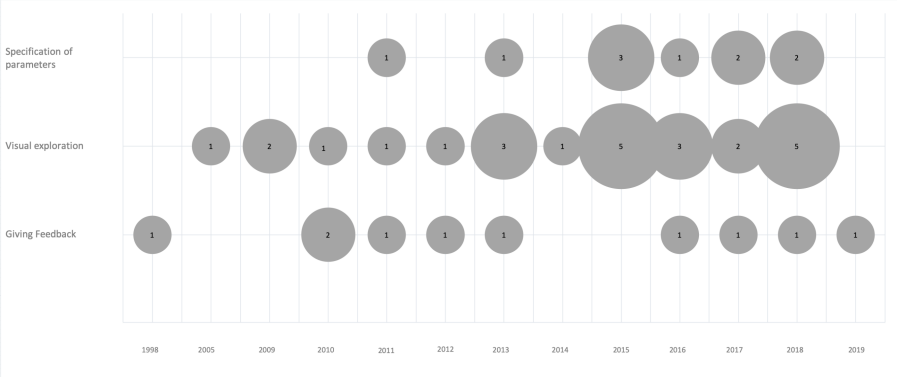

Figure A.12: Type of process is done by a human per year.

information about the origin, venue of publication and publishers of the papers that were selected for our study.

A.2 LDA topics of COVID19 data set

Table A.8: Our LDA topics with top 10 terms in each topic that represent the content of topic.

Topic	Top 10 terms
Topic 1	health, public, country, research, system, disease, new, global, need, state.
Topic 2	cell, activity, target, high, acid, protein, system, peptide, nanoparticle, method
Topic 3	patient, covid-19, study, disease, severe, sars-cov-2, clinical, case, infection, day
Topic 4	1h, hz, compound, 2h, nmr, mhz, ml, h5n1, mmol, yield
Topic 5	covid-19, study, datum, display, author/funder, grant, holder, version, post, perpetuity
Topic 6	der, die, und, von, eine, mit, bei, zu, ist, den
Topic 7	de, la, des, les, le, en, une, dans, du, par
Topic 8	market, cost, model, price, increase, effect, result, economic, study, value
Topic 9	pig, pedv, group, plant, extract, study, piglet, prrsv, day, control,
Topic 10	case, lesion, patient, disease, lung, tumor, cancer, image, diagnosis, ct
Topic 11	health, study, social, participant, student, work, experience, people, covid-19, mental
Topic 12	species, bat, host, study, human, csf, wildlife, olfactory, rabies, sample
Topic 13	sequence, genome, sars-cov-2, read, analysis, tree, species, mutation, alignment, phylogenetic
Topic 14	genetic, mutation, population, variant, allele, gene, genotype, selection, disease, individual
Topic 15	cell, increase, effect, level, protein, receptor, activity, expression, live, tissue
Topic 16	patient, cell, transplant, day, transplantation, disease, donor, recipient, result, year
Topic 17	energy, rate, activity, group, protein, gene, risk, study, human, case
Topic 18	vaccine, antibody, response, vaccination, virus, antigen, immune, mouse, challenge, immunization
Topic 19	child, infection, respiratory, virus, study, influenza, age, year, viral, rsv
Topic 20	group, study, image, patient, result, method, compare, value, mean, high,
Topic 21	water, mask, surface, filter, material, use, air, particle, test, high,
Topic 22	structure, bind, residue, protein, interaction, model, peptide, energy, form, complex
Topic 23	model, case, numb, time, rate, epidemic, population, individual, infect, infection
Topic 24	drug, inhibitor, compound, activity, antiviral, treatment, effect, viral, target, inhibit
Topic 25	patient, increase, pressure, blood, injury, cardiac, failure, ventilation, heart, acute
Topic 26	calve, milk, diarrhea, study, rotavirus, herd, day, group, calf, farm
Topic 27	de, la, en, el, los, que, se, con, las, por
Topic 28	animal, cat, dog, disease, cause, infection, occur, sign, include, horse
Topic 29	lectin, carbohydrate, oligosaccharide, sp-d, glycans, mannose, gp120, grft, sp-a, glycan
Topic 30	rna, sequence, protein, gene, virus, dna, genome, viral, site, codon
Topic 31	patient, care, hospital, covid-19, health, case, disease, infection, pandemic, risk
Topic 32	work, author, article, publication, manuscript, fund, submit, report, journal, research
Topic 33	cell, virus, infection, culture, expression, show, medium, protein, infect, viral
Topic 34	de, van, een, en, het, meet, bij, op, zijn, voor
Topic 35	air, temperature, particle, droplet, concentration, flow, study, model, high, show
Topic 36	cell, immune, response, infection, cytokine, virus, receptor, macrophage, human, mouse
Topic 37	eye, der, ocular, und, retinal, die, corneal, mit, bei, von
Topic 38	datum, information, system, user, process, use, provide, model, example, application
Topic 39	gene, expression, analysis, identify, datum, pathway, cell, study, sample, protein
Topic 40	patient, study, group, treatment, result, datum, clinical, outcome, year, analysis
Topic 41	mouse, lung, infection, cell, response, expression, ace2, viral, virus, level
Topic 42	model, method, datum, network, set, result, algorithm, value, image, train
Topic 43	blood, donor, transfusion, plasma, platelet, product, test, study, risk, cell
Topic 44	virus, strain, human, viral, infection, sequence, host, isolate, infect, coronavirus
Topic 45	protein, cell, viral, membrane, virus, domain, bind, replication, signal, host,
Topic 46	di, la, il, del, che, un, della, con, da, una
Topic 47	sample, test, assay, detection, pcr, positive, detect, result, rna, method
Topic 48	infection, disease, patient, antibiotic, bacterium, bacterial, pathogen, resistance, cause isolate
Topic 49	cell, mouse, response, expression, disease, cd4, brain, cd8, increase, cns
Topic 50	protein, bind, antibody, mm, cell, peptide, contain, buff, serum, show

Bibliography

[1] Stratos Idreos, Olga Papaemmanouil, and Surajit Chaudhuri. Overview of data exploration techniques. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, pages 277–281, 2015. (*Cited on pages 1, 14, 15, and 35.*)

[2] Parantapa Goswami, Eric Gaussier, and Massih-Reza Amini. Exploring the space of information retrieval term scoring functions. *Information Processing & Management*, 53(2):454–472, 2017. (*Cited on pages 1 and 35.*)

[3] Vikram Singh and Ajay Singh. Learn-as-you-go: Feedback-driven result ranking and query refinement for interactive data exploration. *Procedia Computer Science*, 125:550–559, 2018. (*Cited on pages 1 and 35.*)

[4] Kyriaki Dimitriadou, Olga Papaemmanouil, and Yanlei Diao. Aide: an active learning-based approach for interactive data exploration. *IEEE Transactions on Knowledge and Data Engineering*, 28(11):2842–2856, 2016. (*Cited on pages 1, 19, and 35.*)

[5] Magdalini Eirinaki, Suju Abraham, Neoklis Polyzotis, and Naushin Shaikh. Querie: Collaborative database exploration. *IEEE Transactions on knowledge and data engineering*, 26(7):1778–1790, 2013. (*Cited on pages 1, 17, 19, and 35.*)

[6] Marina Drosou and Evaggelia Pitoura. Ymaldb: exploring relational databases via result-driven recommendations. *The VLDB Journal*, 22(6):849–874, 2013. (*Cited on pages 1, 17, 19, and 35.*)

[7] Hao Li, Chee-Yong Chan, and David Maier. Query from examples: An iterative, data-driven approach to query construction. *Proceedings of the VLDB Endowment*, 8(13):2158–2169, 2015. (*Cited on pages 1, 17, 21, and 35.*)

[8] Kyriaki Dimitriadou, Olga Papaemmanouil, and Yanlei Diao. Explore-by-example: An automatic query steering framework for interactive data exploration. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pages 517–528, 2014. (*Cited on pages 1, 17, 21, and 35.*)

[9] Thibault Sellam and Martin Kersten. Cluster-driven navigation of the query space. *IEEE Transactions on Knowledge and Data Engineering*, 28(5):1118–1131, 2016. (*Cited on pages 1, 21, 23, 24, 25, and 35.*)

[10] Carlos Castillo. *Big crisis data: social media in disasters and time-critical situations*. Cambridge University Press, 2016. (*Cited on pages 2 and 3.*)

[11] Ryen W White, Mikhail Bilenko, and Silviu Cucerzan. Studying the use of popular destinations to enhance web search interaction. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 159–166, 2007. (*Cited on page 3.*)

[12] Ricardo Baeza-Yates, Carlos Hurtado, and Marcelo Mendoza. Query recommendation using query logs in search engines. In *International conference on extending database technology*, pages 588–596. Springer, 2004. (*Cited on page 3.*)

[13] Kyriaki Dimitriadou, Olga Papaemmanouil, and Yanlei Diao. Aide: an automated sample-based approach for interactive data exploration. *arXiv preprint arXiv:1510.08897*, 2015. (*Cited on page 4.*)

[14] Matteo Lissandrini, Davide Mottin, Themis Palpanas, and Yannis Velegrakis. Data exploration using example-based methods. *Synthesis Lectures on Data Management*, 10(4):1–164, 2018. (*Cited on pages 9 and 21.*)

[15] John W Tukey et al. *Exploratory data analysis*, volume 2. Reading, Mass., 1977. (*Cited on pages 10 and 14.*)

[16] Marcello Buoncristiano, Giansalvatore Mecca, Elisa Quintarelli, Manuel Roveri, Donatello Santoro, and Letizia Tanca. Database challenges for exploratory computing. *ACM SIGMOD Record*, 44(2):17–22, 2015. (*Cited on pages 10, 14, and 17.*)

[17] Aurélien Saint Requier, Gérard Dupont, Sébastien Adam, and Yves Lecourtier. Évaluation d'outils de reformulation interactive de requêtes.

In *Conférence en Recherche d'Information et Applications*, pages 223–238, 2010. (*Cited on page 11.*)

[18] Jinxi Xu and W Bruce Croft. Quary expansion using local and global document analysis. In *Acm sigir forum*, volume 51, pages 168–175. ACM New York, NY, USA, 2017. (*Cited on page 11.*)

[19] Zhiwei Zhang, Qifan Wang, Luo Si, and Jianfeng Gao. Learning for efficient supervised query expansion via two-stage feature selection. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 265–274, 2016. (*Cited on pages 11 and 12.*)

[20] Daniel Tunkelang. Faceted search. *Synthesis lectures on information concepts, retrieval, and services*, 1(1):1–80, 2009. (*Cited on pages 11 and 12.*)

[21] Alkis Simitsis, Akanksha Baid, Yannis Sismanis, and Berthold Reinwald. Multidimensional content exploration. *Proceedings of the VLDB Endowment*, 1(1):660–671, 2008. (*Cited on pages 11 and 12.*)

[22] Nikos Sarkas, Nilesh Bansal, Gautam Das, and Nick Koudas. Measure-driven keyword-query expansion. *Proceedings of the VLDB Endowment*, 2(1):121–132, 2009. (*Cited on pages 11 and 12.*)

[23] Btihal El Ghali, Abderrahim El Qadi, Omar El Midaoui, Mohamed Ouadou, and Driss Aboutajdine. Probabilistic query expansion method based on a query recommendation algorithm. *Int. J. Web Appl.*, 5(1):1–12, 2013. (*Cited on pages 11 and 13.*)

[24] Martin L Kersten, Stratos Idreos, Stefan Manegold, and Erietta Liarou. The researcher's guide to the data deluge: Querying a scientific database in just a few seconds. *Proceedings of the VLDB Endowment*, 4(12):1474–1477, 2011. (*Cited on pages 11, 13, 22, 32, 33, and 34.*)

[25] Jay Patel and Vikram Singh. Query morphing: A proximity-based approach for data exploration and query reformulation. In *International Conference on Mining Intelligence and Knowledge Exploration*, pages 261–273. Springer, 2017. (*Cited on pages 11 and 13.*)

[26] Lilong Jiang and Arnab Nandi. Snaptoquery: providing interactive feedback during exploratory query specification. *Proceedings of the VLDB Endowment*, 8(11):1250–1261, 2015. (*Cited on pages 11 and 23.*)

[27] Ziv Bar-Yossef and Maxim Gurevich. Mining search engine query logs via suggestion sampling. *Proceedings of the VLDB Endowment*, 1(1):54–65, 2008. (*Cited on pages 11 and 12.*)

[28] Ji-Rong Wen, Jian-Yun Nie, and Hong-Jiang Zhang. Clustering user queries of a search engine. In *Proceedings of the 10th international conference on World Wide Web*, pages 162–168, 2001. (*Cited on pages 11 and 20.*)

[29] Hiteshwar Kumar Azad and Akshay Deepak. Query expansion techniques for information retrieval: a survey. *Information Processing & Management*, 56(5):1698–1735, 2019. (*Cited on page 11.*)

[30] Elena Baralis, Paolo Garza, Elisa Quintarelli, and Letizia Tanca. Answering xml queries by means of data summaries. *ACM Transactions on Information Systems (TOIS)*, 25(3):10–es, 2007. (*Cited on page 14.*)

[31] Viswanath Poosala, Venkatesh Ganti, and Yannis E. Ioannidis. Approximate query answering using histograms. *IEEE Data Eng. Bull.*, 22(4):5–14, 1999. (*Cited on page 14.*)

[32] Prasanth Jayachandran, Karthik Tunga, Niranjan Kamat, and Arnab Nandi. Combining user interaction, speculative query execution and sampling in the dice system. *Proceedings of the VLDB Endowment*, 7(13):1697–1700, 2014. (*Cited on pages 14 and 15.*)

[33] Antonio Giuzio, Giansalvatore Mecca, Elisa Quintarelli, Manuel Roveri, Donatello Santoro, and Letizia Tanca. Indiana: An interactive system for assisting database exploration. *Information Systems*, 83:40–56, 2019. (*Cited on pages 14, 17, and 25.*)

[34] Paolo Bethaz and Tania Cerquitelli. Enhancing the friendliness of data analytics tasks: an automated methodology. In *EDBT/ICDT Workshops*, 2021. (*Cited on pages 14 and 15.*)

[35] Doris Xin, Litian Ma, Jialin Liu, Stephen Macke, Shuchen Song, and Aditya Parameswaran. Helix: accelerating human-in-the-loop machine learning. *arXiv preprint arXiv:1808.01095*, 2018. (*Cited on pages 14 and 15.*)

[36] Udayan Khurana, Srinivasan Parthasarathy, and Deepak S Turaga. Read: Rapid data exploration, analysis and discovery. In *EDBT*, pages 612–615, 2014. (*Cited on pages 14 and 15.*)

[37] Ke Li, Hubert Naacke, and Bernd Amann. Epique: Extracting meaningful science evolution patterns from large document archives (demonstration). In *Int'l Conf. on Extending Database Technology (EDBT). Copenhagen, Denmark*, 2020. (*Cited on pages 14 and 15.*)

[38] Maribel Acosta, Elena Simperl, Fabian Flöck, and Maria-Esther Vidal. Hare: A hybrid sparql engine to enhance query answers via crowdsourcing. In *Proceedings of the 8th International Conference on Knowledge Capture*, pages 1–8, 2015. (*Cited on pages 14 and 15.*)

[39] Tarique Siddiqui, Albert Kim, John Lee, Karrie Karahalios, and Aditya Parameswaran. Effortless data exploration with zenvisage: an expressive and interactive visual analytics system. *arXiv preprint arXiv:1604.03583*, 2016. (*Cited on pages 14 and 16.*)

[40] Christina Christodoulakis, Eser Kandogan, Ignacio G Terrizzano, and Renée J Miller. Viqs: Visual interactive exploration of query semantics. In *Proceedings of the 2017 ACM Workshop on Exploratory Search and Interactive Data Analytics*, pages 25–32, 2017. (*Cited on pages 14 and 16.*)

[41] María Constanza Pabón, Marta Millán, Claudia Roncancio, and César A Collazos. Graphtql: A visual query system for graph databases. *Journal of Computer Languages*, 51:97–111, 2019. (*Cited on pages 14 and 16.*)

[42] Kevin Hu, Diana Orghian, and César Hidalgo. Dive: A mixed-initiative system supporting integrated data exploration workflows. In *Proceedings of the Workshop on Human-In-the-Loop Data Analytics*, pages 1–7, 2018. (*Cited on pages 14, 16, 17, and 20.*)

[43] Kristi Morton, Magdalena Balazinska, Dan Grossman, and Jock Mackinlay. Support the data enthusiast: Challenges for next-generation data-analysis systems. *Proceedings of the VLDB Endowment*, 7(6):453–456, 2014. (*Cited on pages 14, 16, and 17.*)

[44] Pat Hanrahan. Analytic database technologies for a new kind of user: the data enthusiast. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, pages 577–578, 2012. (*Cited on pages 14 and 17.*)

[45] Julien Aligon, Matteo Golfarelli, Patrick Marcel, Stefano Rizzi, and Elisa Turricchia. Similarity measures for olap sessions. *Knowledge*

and information systems, 39(2):463–489, 2014. (*Cited on pages 17, 18, and 19.*)

[46] Gloria Chatzopoulou, Magdalini Eirinaki, Suju Koshy, Sarika Mittal, Neoklis Polyzotis, and Jothi Swarubini Vindhiya Varman. The querie system for personalized query recommendations. *IEEE Data Eng. Bull.*, 34(2):55–60, 2011. (*Cited on pages 17, 18, and 19.*)

[47] Matteo Golfarelli, Stefano Rizzi, and Paolo Biondi. myolap: An approach to express and evaluate olap preferences. *IEEE Transactions on Knowledge and Data Engineering*, 23(7):1050–1064, 2010. (*Cited on pages 17 and 18.*)

[48] Xiaoyan Yang, Cecilia M Procopiuc, and Divesh Srivastava. Recommending join queries via query log analysis. In *2009 IEEE 25th International Conference on Data Engineering*, pages 964–975. IEEE, 2009. (*Cited on pages 17 and 18.*)

[49] Di Yang, Elke A Rundensteiner, and Matthew O Ward. Nugget discovery in visual exploration environments by query consolidation. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 603–612, 2007. (*Cited on pages 17 and 18.*)

[50] Marina Drosou and Evaggelia Pitoura. Redrive: result-driven database exploration through recommendations. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 1547–1552, 2011. (*Cited on pages 17, 18, and 20.*)

[51] Alkis Simitsis, Georgia Koutrika, and Yannis Ioannidis. Précis: from unstructured keywords as queries to structured databases as answers. *The VLDB Journal*, 17(1):117–149, 2008. (*Cited on pages 17, 18, and 23.*)

[52] Romain Perriot, Laurent d'Orazio, Dominique Laurent, and Nicolas Spyratos. A semantic matrix for aggregate query rewriting. In *International Workshop on Information Search, Integration, and Personalization*, pages 46–66. Springer, 2015. (*Cited on pages 17 and 18.*)

[53] Bill Howe, Garret Cole, Nodira Khoussainova, and Leilani Battle. Automatic example queries for ad hoc databases. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*, pages 1319–1322, 2011. (*Cited on pages 17 and 18.*)

[54] Marie Le Guilly, Jean-Marc Petit, and Vasile-Marian Scuturici. Sql query completion for data exploration. *arXiv preprint arXiv:1802.02872*, 2018. (*Cited on pages 17 and 19.*)

[55] Hamada M Zahera, Gamal F El-Hady, and Waiel F Abd El-Wahed. Query recommendation for improving search engine results. In *Information Retrieval Methods for Multidisciplinary Applications*, pages 46–53. IGI Global, 2013. (*Cited on pages 17 and 19.*)

[56] Gloria Chatzopoulou, Magdalini Eirinaki, and Neoklis Polyzotis. Query recommendations for interactive database exploration. In *International Conference on Scientific and Statistical Database Management*, pages 3–18. Springer, 2009. (*Cited on pages 17 and 19.*)

[57] Javad Akbarnejad, Gloria Chatzopoulou, Magdalini Eirinaki, Suju Koshy, Sarika Mittal, Duc On, Neoklis Polyzotis, and Jothi S Vindhiya Varman. Sql querie recommendations. *Proceedings of the VLDB Endowment*, 3(1-2):1597–1600, 2010. (*Cited on pages 17 and 19.*)

[58] Wei Chen, Fangzhou Guo, Dongming Han, Jacheng Pan, Xiaotao Nie, Jiazhi Xia, and Xiaolong Zhang. Structure-based suggestive exploration: a new approach for effective exploration of large networks. *IEEE transactions on visualization and computer graphics*, 25(1):555–565, 2018. (*Cited on pages 17 and 21.*)

[59] Sihem Amer-Yahia, Georgia Koutrika, Frederic Bastian, Theofilos Belmpas, Martin Braschler, Ursin Brunner, Diego Calvanese, Maximilian Fabricius, Orest Gkini, Catherine Kosten, et al. Inode: Building an end-to-end data exploration system in practice [extended vision]. *arXiv preprint arXiv:2104.04194*, 2021. (*Cited on pages 17, 21, 22, and 23.*)

[60] Francesco Ventura, Salvatore Greco, Daniele Apiletti, and Tania Cerquitelli. Explaining the deep natural language processing by mining textual interpretable features. *arXiv preprint arXiv:2106.06697*, 2021. (*Cited on page 17.*)

[61] Jingwei Zuo, Karine Zeitouni, and Yehia Taher. Incremental and adaptive feature exploration over time series stream. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 593–602. IEEE, 2019. (*Cited on pages 17 and 18.*)

[62] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings*

of the 10th international conference on World Wide Web, pages 285–295, 2001. (*Cited on page 19.*)

[63] Bruno M Fonseca, Paulo Braz Golgher, Edleno Silva de Moura, and Nivio Ziviani. Using association rules to discover search engines related queries. In *Proceedings of the IEEE/LEOS 3rd International Conference on Numerical Simulation of Semiconductor Optoelectronic Devices (IEEE Cat. No. 03EX726)*, pages 66–71. IEEE, 2003. (*Cited on page 20.*)

[64] Nelly Vouzoukidou, Bernd Amann, and Vassilis Christophides. Continuous top-k queries over real-time web streams. *arXiv preprint arXiv:1610.06500*, 2016. (*Cited on page 20.*)

[65] Nelly Vouzoukidou, Bernd Amann, and Vassilis Christophides. Meowsreader: Real-time ranking and filtering of news with generalized continuous top-k queries. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 2066–2068, 2014. (*Cited on page 20.*)

[66] Ugur Cetintemel, Mitch Cherniack, Justin DeBrabant, Yanlei Diao, Kyriaki Dimitriadou, Alexander Kalinin, Olga Papaemmanouil, and Stanley B Zdonik. Query steering for interactive data exploration. In *CIDR*, 2013. (*Cited on page 23.*)

[67] Sanjay Agrawal, Surajit Chaudhuri, and Gautam Das. Dbxplorer: enabling keyword search over relational databases. In *Proceedings of the 2002 ACM SIGMOD international conference on Management of data*, pages 627–627, 2002. (*Cited on page 23.*)

[68] Thibault Sellam and Martin Kersten. Ziggy: Characterizing query results for data explorers. *Proceedings of the VLDB Endowment*, 9(13):1473–1476, 2016. (*Cited on page 23.*)

[69] Daniel Deutch and Amir Gilad. Qplain: Query by explanation. In *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*, pages 1358–1361. IEEE, 2016. (*Cited on pages 23 and 24.*)

[70] Melanie Herschel, Ralf Diestelkämper, and Houssem Ben Lahmar. A survey on provenance: What for? what form? what from? *The VLDB Journal*, 26(6):881–906, 2017. (*Cited on page 24.*)

[71] Todd J Green. Containment of conjunctive queries on annotated relations. *Theory of Computing Systems*, 49(2):429–459, 2011. (*Cited on page 24.*)

[72] Jayavel Shanmugasundaram, Jerry Kiernan, Eugene J Shekita, Catalina Fan, and John Funderburk. Querying xml views of relational data. In *VLDB*, volume 1, pages 261–270, 2001. (*Cited on page 28.*)

[73] Marcos Roberto Ribeiro, Maria Camila N Barioni, Sandra de Amo, Claudia Roncancio, and Cyril Labbé. Incremental evaluation of continuous preference queries. *Information Sciences*, 453:127–153, 2018. (*Cited on page 30.*)

[74] Ludovico Boratto, Salvatore Carta, Alessandro Chessa, Maurizio Agelli, and M Laura Clemente. Group recommendation with automatic identification of users communities. In *2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*, volume 3, pages 547–550. IEEE, 2009. (*Cited on page 32.*)

[75] Sihem Amer-Yahia, Senjuti Basu Roy, Ashish Chawlat, Gautam Das, and Cong Yu. Group recommendation: Semantics and efficiency. *Proceedings of the VLDB Endowment*, 2(1):754–765, 2009. (*Cited on page 32.*)

[76] Mark O'Connor, Dan Cosley, Joseph A Konstan, and John Riedl. Polylens: a recommender system for groups of users. In *ECSCW 2001*, pages 199–218. Springer, 2001. (*Cited on page 32.*)

[77] Sihem Amer-Yahia, Shady Elbassuoni, Behrooz Omidvar-Tehrani, Ria Borromeo, and Mehrdad Farokhnejad. Grouptravel: Customizing travel packages for groups. 2019. (*Cited on page 33.*)

[78] Joseph Rocchio. Relevance feedback in information retrieval. *The Smart retrieval system-experiments in automatic document processing*, pages 313–323, 1971. (*Cited on page 38.*)

[79] Yashen Wang, Heyan Huang, and Chong Feng. Query expansion based on a feedback concept model for microblog retrieval. In *Proceedings of the 26th International Conference on World Wide Web*, pages 559–568, 2017. (*Cited on page 38.*)

[80] Chengxiang Zhai and John Lafferty. Model-based feedback in the language modeling approach to information retrieval. In *Proceedings*

of the tenth international conference on Information and knowledge management, pages 403–410, 2001. (*Cited on page 38.*)

[81] Peter D Turney and Patrick Pantel. From frequency to meaning: Vector space models of semantics. *Journal of artificial intelligence research*, 37:141–188, 2010. (*Cited on page 39.*)

[82] Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. *Introduction to information retrieval*, volume 39. Cambridge University Press Cambridge, 2008. (*Cited on page 39.*)

[83] Tomáš Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies*, pages 746–751, 2013. (*Cited on page 39.*)

[84] Ellen M Voorhees. Query expansion using lexical-semantic relations. In *SIGIR'94*, pages 61–69. Springer, 1994. (*Cited on page 41.*)

[85] Alan F Smeaton, Fergus Kelledy, and Ruairi O'Donnell. Trec-4 experiments at dublin city university: Thresholding posting lists, query expansion with wordnet and pos tagging of spanish. *Harman [6]*, pages 373–389, 1995. (*Cited on page 41.*)

[86] Dwaipayan Roy, Debjyoti Paul, Mandar Mitra, and Utpal Garain. Using word embeddings for automatic query expansion. *arXiv preprint arXiv:1606.07608*, 2016. (*Cited on page 42.*)

[87] Fernando Diaz, Bhaskar Mitra, and Nick Craswell. Query expansion with locally-trained word embeddings. *arXiv preprint arXiv:1605.07891*, 2016. (*Cited on page 42.*)

[88] Youngho Kim, Jangwon Seo, and W Bruce Croft. Automatic boolean query suggestion for professional search. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 825–834, 2011. (*Cited on pages 42 and 77.*)

[89] Evgeniy Gabrilovich, Andrei Broder, Marcus Fontoura, Amruta Joshi, Vanja Josifovski, Lance Riedel, and Tong Zhang. Classifying search queries using the web as a source of knowledge. *ACM Transactions on the Web (TWEB)*, 3(2):1–28, 2009. (*Cited on page 59.*)

[90] Collin McMillan, Mark Grechanik, Denys Poshyvanyk, Qing Xie, and Chen Fu. Portfolio: finding relevant functions and their usage. In *Proceedings of the 33rd International Conference on Software Engineering*, pages 111–120, 2011. (*Cited on page 60.*)

[91] Wing-Kwan Chan, Hong Cheng, and David Lo. Searching connected api subgraph via text phrases. In *Proceedings of the ACM SIGSOFT 20th International Symposium on the Foundations of Software Engineering*, pages 1–11, 2012. (*Cited on page 60.*)

[92] Ferdian Thung, Shaowei Wang, David Lo, and Julia Lawall. Automatic recommendation of api methods from feature requests. In *2013 28th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 290–300. IEEE, 2013. (*Cited on page 60.*)

[93] Moumita Basu, Anurag Shandilya, Prannay Khosla, Kripabandhu Ghosh, and Saptarshi Ghosh. Extracting resource needs and availabilities from microblogs for aiding post-disaster relief operations. *IEEE Transactions on Computational Social Systems*, 6(3):604–618, 2019. (*Cited on page 60.*)

[94] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013. (*Cited on pages 74 and 98.*)

[95] Raj Ratn Pranesh, Mehrdad Farokhnejad, Ambesh Shekhar, and Genoveva Vargas-Solar. Looking for covid-19 misinformation in multilingual social media texts. *arXiv preprint arXiv:2105.03313*, 2021. (*Cited on page 74.*)

[96] IC Mogotsi. Christopher d. manning, prabhakar raghavan, and hinrich schütze: Introduction to information retrieval, 2010. (*Cited on page 74.*)

[97] Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, et al. Okapi at trec-3. *Nist Special Publication Sp*, 109:109, 1995. (*Cited on page 74.*)

[98] Quoc Trung Tran, Chee-Yong Chan, and Srinivasan Parthasarathy. Query reverse engineering. *The VLDB Journal*, 23(5):721–746, 2014. (*Cited on page 77.*)

[99] Richard Bache and Leif Azzopardi. Improving access to large patent corpora. In *Transactions on large-scale data-and knowledge-centered systems II*, pages 103–121. Springer, 2010. (*Cited on page 77.*)

[100] John F Pane and Brad A Myers. Improving user performance on boolean queries. In *CHI'00 Extended Abstracts on Human Factors in Computing Systems*, pages 269–270, 2000. (*Cited on page 77.*)

[101] W Xindong, J Vipin Kumar, R Quinlan, J Ghosh, Q Yang, H Motoda, GJ McLachlan, N Angus, Bing Liu, S Philip, et al. Top 10 algorithms in data mining. knowledge and information systems. 2008. (*Cited on page 77.*)

[102] Yosi Benasher and Ilan Newman. Decision trees with boolean threshold queries. *Journal of Computer and System Sciences*, 51(3):495–502, 1995. (*Cited on page 77.*)

[103] Stuart Russell and Peter Norvig. Artificial intelligence: a modern approach. 2002. (*Cited on page 77.*)

[104] Richard Bache. Measuring and improving access to the corpus. In *Current Challenges in Patent Information Retrieval*, pages 147–165. Springer, 2011. (*Cited on page 80.*)

[105] Mandar Mitra, Amit Singhal, and Chris Buckley. Improving automatic query expansion. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 206–214, 1998. (*Cited on page 80.*)

[106] RA Devi and K Nirmala. Construction of decision tree: Attribute selection measures. *International Journal of Advancements in Research & Technology*, 2(4):343–347, 2013. (*Cited on page 85.*)

[107] Muhammad Imran, Prasenjit Mitra, and Carlos Castillo. Twitter as a lifeline: Human-annotated twitter corpora for nlp of crisis-related messages. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, Paris, France, may 2016. European Language Resources Association (ELRA). (*Cited on page 90.*)

[108] 2020 Poynter Institute. The international fact-checking network., 2020. (*Cited on pages 91 and 92.*)

[109] Firoj Alam, Shaden Shaar, Fahim Dalvi, Hassan Sajjad, Alex Nikolov, Hamdy Mubarak, Giovanni Da San Martino, Ahmed Abdelali, Nadir Durrani, Kareem Darwish, and Preslav Nakov. Fighting the covid-19 infodemic: Modeling the perspective of journalists, fact-checkers, social media platforms, policy makers, and the society, 2020. (*Cited on page 92.*)

[110] Emily Chen, Kristina Lerman, and Emilio Ferrara. Tracking social media discourse about the covid-19 pandemic: Development of a public coronavirus twitter data set. *JMIR Public Health and Surveillance*, 6(2):e19273, 2020. (*Cited on page 92.*)

[111] Charin Polpanumas Arthit Suriyawongkul Lalita Lowphansirikul Pattarawat Chormai Wannaphong Phatthiyaphaibun, Korakot Chaovavanich. PyThaiNLP: Thai Natural Language Processing in Python, June 2016. (*Cited on page 93.*)

[112] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016. (*Cited on page 95.*)

[113] Lucy Lu Wang, Kyle Lo, Yoganand Chandrasekhar, Russell Reas, Jiangjiang Yang, Darrin Eide, Kathryn Funk, Rodney Kinney, Ziyang Liu, William Merrill, et al. Cord-19: The covid-19 open research dataset. *arXiv preprint arXiv:2004.10706*, 2020. (*Cited on pages 96 and 97.*)

[114] Radim Řehůřek and Petr Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA. http://is.muni.cz/publication/884893/en. (*Cited on page 98.*)

[115] Jimmy Lin. Is searching full text more effective than searching abstracts? *BMC bioinformatics*, 10(1):46, 2009. (*Cited on page 99.*)

[116] Mark Neumann, Daniel King, Iz Beltagy, and Waleed Ammar. Scispacy: Fast and robust models for biomedical natural language processing. *arXiv preprint arXiv:1902.07669*, 2019. (*Cited on page 99.*)

[117] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003. (*Cited on page 99.*)

[118] Katharine E Hubbard and Sonja D Dunbar. Perceptions of scientific research literature and strategies for reading papers depend on academic career stage. *PloS one*, 12(12), 2017. (*Cited on page 106.*)

[119] Jinwei Ai, Junwen Chen, Yong Wang, Xiaoyun Liu, Wufeng Fan, Gaojing Qu, Meiling Zhang, Shengduo Polo Pei, Bowen Tang, Shuai Yuan, et al. The cross-sectional study of hospitalized coronavirus disease 2019 patients in xiangyang, hubei province. *MedRxiv*, 2020. (*Cited on page 119.*)

[120] Gil Caspi, Uri Shalit, Soren Lund Kristensen, Doron Aronson, Lilac Caspi, Oran Rossenberg, Avi Shina, and Oren Caspi. Climate effect on covid-19 spread rate: an online surveillance tool. *medRxiv*, 2020. (*Cited on pages 110 and 119.*)

[121] Andrew Clark, Mark Jit, Charlotte Warren-Gash, Bruce Guthrie, Harry HX Wang, Stewart W Mercer, Colin Sanderson, Martin McKee, Christopher Troeger, Kanyin I Ong, et al. How many are at increased risk of severe covid-19 disease? rapid global, regional and national estimates for 2020. *medRxiv*, 2020. (*Cited on page 119.*)

[122] Fatima Amanat and Florian Krammer. Sars-cov-2 vaccines: status report. *Immunity*, 2020. (*Cited on page 119.*)

[123] Jia Wang and Zhifeng Wang. Strengths, weaknesses, opportunities and threats (swot) analysis of china's prevention and control strategy for the covid-19 epidemic. *International Journal of Environmental Research and Public Health*, 17(7):2235, 2020. (*Cited on page 119.*)

[124] Samuli Laato, AKM Islam, Muhammad Nazrul Islam, and Eoin Whelan. Why do people share misinformation during the covid-19 pandemic? *arXiv preprint arXiv:2004.09600*, 2020. (*Cited on page 119.*)

[125] Viktor Stojkoski, Zoran Utkovski, Petar Jolakoski, Dragan Tevdovski, and Ljupco Kocarev. The socio-economic determinants of the coronavirus disease (covid-19) pandemic. *arXiv preprint arXiv:2004.07947*, 2020. (*Cited on page 119.*)

[126] Bo Diao, Kun Wen, Jian Chen, Yueping Liu, Zilin Yuan, Chao Han, Jiahui Chen, Yuxian Pan, Li Chen, Yunjie Dan, et al. Diagnosis of

acute respiratory syndrome coronavirus 2 infection by detection of nucleocapsid protein. *medRxiv*, 2020. (*Cited on page 119.*)

[127] Ioanna Smyrlaki, Martin Ekman, Martin Vondracek, Natali Papanicoloau, Antonio Lentini, Johan Aarum, Shaman Muradrasoli, Jan Albert, Björn Högberg, and Björn Reinius. Massive and rapid covid-19 testing is feasible by extraction-free sars-cov-2 rt-qpcr. *medRxiv*, 2020. (*Cited on page 119.*)

[128] Matthew Levin, Martin D Chen, Anjan Shah, Ronak Shah, George Zhou, Erica Kane, Garrett Burnett, Shams Ranginwala, Jonathan Madek, Christopher Gidiscin, et al. Differential ventilation using flow control valves as a potential bridge to full ventilatory support during the covid-19 crisis. *medRxiv*, 2020. (*Cited on page 119.*)

[129] Emanuel Goldman. Exaggerated risk of transmission of covid-19 by fomites. *The Lancet Infectious Diseases*, 20(8):892–893, 2020. (*Cited on page 120.*)

[130] Caroline X Gao, Yuguo Li, Jianjian Wei, Sue Cotton, Matthew Hamilton, Lei Wang, and Benjamin J Cowling. Multi-route respiratory infection: when a transmission route may dominate. *medRxiv*, 2020. (*Cited on page 120.*)

[131] Paul-Henri Lambert, Donna M Ambrosino, Svein R Andersen, Ralph S Baric, Steven B Black, Robert T Chen, Cornelia L Dekker, Arnaud M Didierlaurent, Barney S Graham, Samantha D Martin, et al. Consensus summary report for cepi/bc march 12-13, 2020 meeting: Assessment of risk of disease enhancement with covid-19 vaccines. *Vaccine*, 2020. (*Cited on page 120.*)

[132] Vanessa Aparecida Marcolino, Tatiana Colombo Pimentel, and Carlos Eduardo Barão. What to expect from different drugs used in the treatment of covid-19: A study on applications and in vivo and in vitro results. *European Journal of Pharmacology*, 887:173467, 2020. (*Cited on page 120.*)

[133] Lindsay Kim, Shikha Garg, Alissa O'Halloran, Michael Whitaker, Huong Pham, Evan J Anderson, Isaac Armistead, Nancy M Bennett, Laurie Billing, Kathryn Como-Sabetti, et al. Risk factors for intensive care unit admission and in-hospital mortality among hospitalized adults identified through the us coronavirus disease 2019 (covid-19)-associated

hospitalization surveillance network (covid-net). *Clinical Infectious Diseases*, 2020. (*Cited on page 120.*)

[134] Jennifer M Reckrey. Covid-19 confirms it: Paid caregivers are essential members of the healthcare team. *Journal of the American Geriatrics Society*, 2020. (*Cited on page 120.*)

[135] Alexandra Martin, Alexandre Storto, Barbara Andre, Allison Mallory, Remi Dangla, Benoit Visseaux, and Olivier Gossner. High-sensitivity covid-19 group testing by digital pcr. *arXiv preprint arXiv:2006.02908*, 2020. (*Cited on page 120.*)

[136] Zi-Wei Ye, Shuofeng Yuan, Kit-San Yuen, Sin-Yee Fung, Chi-Ping Chan, and Dong-Yan Jin. Zoonotic origins of human coronaviruses. *International journal of biological sciences*, 16(10):1686, 2020. (*Cited on page 120.*)

[137] Julien Arino, Nicolas Bajeux, Stephanie Portet, and James Watmough. Assessing the risk of covid-19 importation and the effect of quarantine. *medRxiv*, 2020. (*Cited on page 120.*)

[138] Eduardo Massad, Marcos Amaku, Annelies Wilder-Smith, Paulo Cesar Costa dos Santos, Claudio Jose Struchiner, and Francisco Antonio Bezerra Coutinho. Two complementary model-based methods for calculating the risk of international spreading of anovel virus from the outbreak epicentre. the case of covid-19. *Epidemiology & Infection*, pages 1–19, 2020. (*Cited on page 120.*)

[139] Leonardo Stella, Alejandro Pinel Martínez, Dario Bauso, and Patrizio Colaneri. The role of asymptomatic individuals in the covid-19 pandemic via complex networks. *arXiv preprint arXiv:2009.03649*, 2020. (*Cited on page 120.*)

[140] Shu Yuan, S Jiang, Zi-Lin Li, et al. Do humidity and temperature impact the spread of the novel coronavirus? *Frontiers in Public Health*, 8:240, 2020. (*Cited on page 120.*)

[141] Hatem A Elshabrawy. Sars-cov-2: An update on potential antivirals in light of sars-cov antiviral drug discoveries. *Vaccines*, 8(2):335, 2020. (*Cited on page 120.*)

[142] Hui Poh Goh, Wafiah Ilyani Mahari, Norhadyrah Izazie Ahad, Liling Chaw, Nurolaini Kifli, Bey Hing Goh, Siang Fei Yeoh, and Long Chiau

Ming. Risk factors affecting covid-19 case fatality rate: A quantitative analysis of top 50 affected countries. *medRxiv*, 2020. (*Cited on page 120.*)

[143] Eugene Merzon, Dmitry Tworowski, Alessandro Gorohovski, Shlomo Vinker, Avivit Golan Cohen, Ilan Green, and Milana Frenkel-Morgenstern. Low plasma 25 (oh) vitamin d level is associated with increased risk of covid-19 infection: an israeli population-based study. *The FEBS journal*, 287(17):3693–3702, 2020. (*Cited on page 120.*)

[144] Yichun Cheng, Ran Luo, Kun Wang, Meng Zhang, Zhixiang Wang, Lei Dong, Junhua Li, Ying Yao, Shuwang Ge, and Gang Xu. Kidney disease is associated with in-hospital death of patients with covid-19. *Kidney international*, 2020. (*Cited on page 120.*)

[145] Ya Gao, Ming Liu, Shuzhen Shi, Yamin Chen, Yue Sun, Ji Chen, and Jinhui Tian. Cancer is associated with the severity and mortality of patients with covid-19: a systematic review and meta-analysis. *medRxiv*, 2020. (*Cited on page 120.*)

[146] Patrick Zimmerman, Stephanie Stroever, Timothy Burton, Karri Hester, Minha Kim, Ryan Fahy, Kimberly Corbitt, Joann Petrini, and Jeffrey Nicastro. Mortality associated with intubation and mechanical ventilation in patients with covid-19. *medRxiv*, 2020. (*Cited on page 120.*)

[147] Xue Li, Wei Xu, Marshall Dozier, Yazhou He, Amir Kirolos, Evropi Theodoratou, et al. The role of children in transmission of sars-cov-2: A rapid review. *Journal of global health*, 10(1), 2020. (*Cited on page 120.*)

[148] Zhimin Chen, Lin Tong, Yunlian Zhou, Chunzhen Hua, Wei Wang, Junfen Fu, Qiang Shu, Liang Hong, Huiqing Xu, Zhen Xu, et al. Childhood covid-19: a multicentre retrospective study. *Clinical Microbiology and Infection*, 26(9):1260–e1, 2020. (*Cited on page 120.*)

[149] Stephen M Kissler, Christine Tedijanto, Edward Goldstein, Yonatan H Grad, and Marc Lipsitch. Projecting the transmission dynamics of sars-cov-2 through the postpandemic period. *Science*, 368(6493):860–868, 2020. (*Cited on page 110.*)

[150] Ermengol Coma, Nuria Mora, Albert Prats-Uribe, Francesc Fina, Daniel Prieto-Alhambra, and Manuel Medina-Peralta. Excess cases of influenza suggest an earlier start to the coronavirus epidemic in

spain than official figures tell us: an analysis of primary care electronic medical records from over 6 million people from catalonia. *medRxiv*, 2020. (*Cited on page 110.*)

[151] Sheera Frenkel, Davey Alba, and Raymond Zhong. Surge of virus misinformation stumps facebook and twitter. *The New York Times*, 2020. (*Cited on page 112.*)

[152] Matteo Cinelli, Walter Quattrociocchi, Alessandro Galeazzi, Carlo Michele Valensise, Emanuele Brugnoli, Ana Lucia Schmidt, Paola Zola, Fabiana Zollo, and Antonio Scala. The covid-19 social media infodemic. *arXiv preprint arXiv:2003.05004*, 2020. (*Cited on page 112.*)

[153] Gunther Eysenbach, John Powell, Oliver Kuss, and Eun-Ryoung Sa. Empirical studies assessing the quality of health information for consumers on the world wide web: a systematic review. *Jama*, 287(20):2691–2700, 2002. (*Cited on page 112.*)

[154] Peter Hernon. Disinformation and misinformation through the internet: Findings of an exploratory study. *Government information quarterly*, 12(2):133–139, 1995. (*Cited on page 112.*)

[155] Gautam Kishore Shahi and Durgesh Nandini. Fakecovid–a multilingual cross-domain fact check news dataset for covid-19. *arXiv preprint arXiv:2006.11343*, 2020. (*Cited on page 112.*)

[156] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. (*Cited on page 113.*)

[157] Kai Petersen, Robert Feldt, Shahid Mujtaba, and Michael Mattsson. Systematic mapping studies in software engineering. In *Ease*, volume 8, pages 68–77, 2008. (*Cited on pages 129 and 132.*)

[158] Mourad Ouzzani, Hossam Hammady, Zbys Fedorowicz, and Ahmed Elmagarmid. Rayyan—a web and mobile app for systematic reviews. *Systematic reviews*, 5(1):210, 2016. (*Cited on pages 131 and 132.*)

[159] Chia-Lung Wu and Yi-Chang Chiang. A geodesign framework procedure for developing flood resilient city. *Habitat international*, 75:78–89, 2018. (*Cited on pages 133 and 134.*)

[160] Mingzhao Li, Zhifeng Bao, Farhana Choudhury, and Timos Sellis. Interactive visualization of urban areas of interest: A parameter-free and efficient footprint method. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 782–785. ACM, 2019. (*Cited on pages 133 and 134.*)

[161] Wei Zeng, Chi-Wing Fu, Stefan Müller Arisona, Simon Schubiger, Remo Burkhard, and Kwan-Liu Ma. A visual analytics design for studying crowd movement rhythms from public transportation data. In *SIGGRAPH ASIA 2016 Symposium on Visualization*, page 4. ACM, 2016. (*Cited on pages 133 and 134.*)

[162] Huajian Mao, Wuman Luo, Haoyu Tan, Lionel M Ni, and Nong Xiao. Exploration of ground truth from raw gps data. In *Proceedings of the ACM SIGKDD International Workshop on Urban Computing*, pages 118–125. ACM, 2012. (*Cited on pages 133 and 134.*)

[163] Yang Wang, Tao Mei, Jingdong Wang, Houqiang Li, and Shipeng Li. Jigsaw: interactive mobile visual search with multimodal queries. In *Proceedings of the 19th ACM international conference on Multimedia*, pages 73–82. ACM, 2011. (*Cited on pages 133 and 134.*)

[164] Chandan Kumar, Wilko Heuten, and Susanne Boll. A visual interactive system for spatial querying and ranking of geographic regions. In *Proceedings of the 13th International Conference on Knowledge Management and Knowledge Technologies*, page 30. ACM, 2013. (*Cited on pages 133 and 134.*)

[165] Michael J Cafarella and Oren Etzioni. A search engine for natural language applications. In *Proceedings of the 14th international conference on World Wide Web*, pages 442–452. ACM, 2005. (*Cited on pages 133 and 134.*)

[166] Cheng-Te Li, Man-Kwan Shan, and Shou-De Lin. Context-based people search in labeled social networks. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 1607–1612. ACM, 2011. (*Cited on pages 133 and 134.*)

[167] Blake Shaw, Jon Shea, Siddhartha Sinha, and Andrew Hogue. Learning to rank for spatiotemporal search. In *Proceedings of the sixth ACM international conference on Web search and data mining*, pages 717–726. ACM, 2013. (*Cited on pages 133 and 134.*)

[168] Madhur Behl and Rahul Mangharam. Interactive analytics for smart cities infrastructures. In *2016 1st International Workshop on Science of Smart City Operations and Platforms Engineering (SCOPE) in partnership with Global City Teams Challenge (GCTC)(SCOPE-GCTC)*, pages 1–6. IEEE, 2016. (*Cited on pages 133 and 134.*)

[169] Zijian Ma, Dan Lu, Qian Liu, Jingyuan Wang, and Zhang Xiong. Cityeyes: A multi-source data integration basec smart city analysis system. In *2017 IEEE 18th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pages 1–3. IEEE, 2017. (*Cited on pages 133 and 134.*)

[170] İbrahim Kök, Mehmet Ulvi Şimşek, and Suat Özdemir. A deep learning model for air quality prediction in smart cities. In *2017 IEEE International Conference on Big Data (Big Data)*, pages 1983–1990. IEEE, 2017. (*Cited on pages 133 and 134.*)

[171] Tao Tang, Xiangjie Kong, Menglin Li, Jinzhong Wang, Guojiang Shen, and Xinshuang Wang. Visos: A visual interactive system for spatial-temporal exploring station importance based on subway data. *IEEE Access*, 6:42131–42141, 2018. (*Cited on pages 133 and 134.*)

[172] Erica Rosalina, Flora D Salim, and Timos Sellis. Automated density-based clustering of spatial urban data for interactive data exploration. In *2017 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 295–300. IEEE, 2017. (*Cited on pages 133 and 134.*)

[173] Jan Nykl, Michal Jakob, and Jan Hrncir. Efficient fine-grained analysis of urban transport accessibility. In *2015 Smart Cities Symposium Prague (SCSP)*, pages 1–5. IEEE, 2015. (*Cited on pages 133 and 134.*)

[174] Alireza Karduni, Isaac Cho, Ginette Wessel, William Ribarsky, Eric Sauda, and Wenwen Dou. Urban space explorer: a visual analytics system for urban planning. *IEEE computer graphics and applications*, 37(5):50–60, 2017. (*Cited on pages 133 and 134.*)

[175] Alexander Bock, Alexander Kleiner, Jonas Lundberg, and Timo Ropinski. An interactive visualization system for urban search & rescue mission planning. In *2014 IEEE International Symposium on Safety, Security, and Rescue Robotics (2014)*, pages 1–7. IEEE, 2014. (*Cited on pages 133 and 134.*)

[176] Qiaomu Shen, Wei Zeng, Yu Ye, Stefan Müller Arisona, Simon Schubiger, Remo Burkhard, and Huamin Qu. Streetvizor: Visual exploration of human-scale urban forms based on street views. *IEEE transactions on visualization and computer graphics*, 24(1):1004–1013, 2017. (*Cited on pages 133 and 134.*)

[177] Harish Doraiswamy, Huy T Vo, Cláudio T Silva, and Juliana Freire. A gpu-based index to support interactive spatio-temporal queries over historical data. In *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*, pages 1086–1097. IEEE, 2016. (*Cited on pages 133 and 134.*)

[178] Wenchao Wu, Yixian Zheng, Nan Cao, Haipeng Zeng, Bing Ni, Huamin Qu, and Lionel M Ni. Mobiseg: Interactive region segmentation using heterogeneous mobility data. In *2017 IEEE Pacific Visualization Symposium (PacificVis)*, pages 91–100. IEEE, 2017. (*Cited on pages 133 and 134.*)

[179] Fang Wen and Xiaoou Tang. User intention modeling for interactive image retrieval, May 29 2012. US Patent 8,190,604. (*Cited on pages 133 and 134.*)

[180] Zuchao Wang, Min Lu, Xiaoru Yuan, Junping Zhang, and Huub Van De Wetering. Visual traffic jam analysis based on trajectory data. *IEEE transactions on visualization and computer graphics*, 19(12):2159–2168, 2013. (*Cited on pages 133 and 134.*)

[181] Steven Luis, Fausto C Fleites, Yimin Yang, Hsin-Yu Ha, and Shu-Ching Chen. A visual analytics multimedia mobile system for emergency response. In *2011 IEEE International Symposium on Multimedia*, pages 337–338. IEEE, 2011. (*Cited on pages 133 and 134.*)

[182] Nivan Ferreira, Jorge Poco, Huy T Vo, Juliana Freire, and Cláudio T Silva. Visual exploration of big spatio-temporal urban data: A study of new york city taxi trips. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2149–2158, 2013. (*Cited on pages 133 and 134.*)

[183] Stefan Hertel, Matthias Wagner, and Rudiger Westermann. A system for visualizing spatiotemporal urban datasets. In *2009 17th International Conference on Geoinformatics*, pages 1–6. IEEE, 2009. (*Cited on pages 133 and 134.*)

[184] Tatiana Von Landesberger, Felix Brodkorb, Philipp Roskosch, Natalia Andrienko, Gennady Andrienko, and Andreas Kerren. Mobilitygraphs: Visual analysis of mass mobility dynamics via spatio-temporal graphs and clustering. *IEEE transactions on visualization and computer graphics*, 22(1):11–20, 2015. (*Cited on pages 133 and 134.*)

[185] Mohammadhossein Ghahramani, MengChu Zhou, and Chi Tin Hon. Spatio-temporal analysis of mobile phone data for interaction recognition. In *2018 IEEE 15th International Conference on Networking, Sensing and Control (ICNSC)*, pages 1–6. IEEE, 2018. (*Cited on pages 133 and 134.*)

[186] Ali Benssam, Nadia Nouali-Taboudjemat, Omar Nouali, and Abdelbaset Kabou. A middleware platform for decision support in disaster management. In *2017 4th International Conference on Information and Communication Technologies for Disaster Management (ICT-DM)*, pages 1–8. IEEE, 2017. (*Cited on pages 133 and 134.*)

[187] Amelia Yzaguirre, Robert Warren, and Mike Smit. Detecting environmental disasters in digital news archives. In *2015 IEEE International Conference on Big Data (Big Data)*, pages 2027–2035. IEEE, 2015. (*Cited on pages 133 and 134.*)

[188] Siyuan Liu, Ce Liu, Qiong Luo, Lionel M Ni, and Huamin Qu. A visual analytics system for metropolitan transportation. In *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 477–480. ACM, 2011. (*Cited on pages 133 and 134.*)

[189] Reza Hassanzadeh, Zorica Nedović-Budić, Akbar Alavi Razavi, Mohsen Norouzzadeh, and Hassan Hodhodkian. Interactive approach for gis-based earthquake scenario development and resource estimation (karmania hazard model). *Computers & geosciences*, 51:324–338, 2013. (*Cited on pages 133 and 134.*)

[190] Liang Yu, Wei Wu, Xiaohui Li, Guangxia Li, Wee Siong Ng, See-Kiong Ng, Zhongwen Huang, Anushiya Arunan, and Hui Min Watt. iviztrans: Interactive visual learning for home and work place detection from massive public transportation data. In *2015 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 49–56. IEEE, 2015. (*Cited on pages 133 and 134.*)

[191] Katerina Vrotsou, Jimmy Johansson, and Matthew Cooper. Activitree: Interactive visual exploration of sequences in event-based data using graph similarity. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):945–952, 2009. (*Cited on pages 133 and 134.*)

[192] Giusy Di Lorenzo, Marco Sbodio, Francesco Calabrese, Michele Berlingerio, Fabio Pinelli, and Rahul Nair. Allaboard: Visual exploration of cellphone mobility data to optimise public transport. *IEEE transactions on visualization and computer graphics*, 22(2):1036–1050, 2015. (*Cited on pages 133 and 134.*)

[193] Ilias Kalamaras, Alexandros Zamichos, Athanasios Salamanis, Anastasios Drosou, Dionysios D Kehagias, Georgios Margaritis, Stavros Papadopoulos, and Dimitrios Tzovaras. An interactive visual analytics platform for smart intelligent transportation systems management. *IEEE Transactions on Intelligent Transportation Systems*, 19(2):487–496, 2017. (*Cited on pages 133 and 134.*)

[194] Mirko Marras, Matteo Manca, Ludovico Boratto, Gianni Fenu, and David Laniado. Barcelonanow: Empowering citizens with interactive dashboards for urban data exploration. In *Companion Proceedings of the The Web Conference 2018*, pages 219–222. International World Wide Web Conferences Steering Committee, 2018. (*Cited on pages 133 and 134.*)

[195] Wu Jiayu, Fu Zhiyong, Liu Zhiyuan, Lin Xu, Tang Jiayu, Pan Jiajia, and Zhao Chen. Creating reflections in public emotion visualization: prototype exploration on traffic theme. In *Proceedings of the 9th ACM Conference on Creativity & Cognition*, pages 357–361. ACM, 2013. (*Cited on pages 133 and 134.*)

[196] Manas Joglekar, Hector Garcia-Molina, and Aditya Parameswaran. Smart drill-down: A new data exploration operator. *Proceedings of the VLDB Endowment*, 8(12):1928–1931, 2015. (*Cited on pages 133 and 134.*)

[197] Andreas Dietze, Marcel Klomann, Yvonne Jung, Michael Englert, Sebastian Rieger, Achim Rehberger, Silvan Hau, and Paul Grimm. Smulgras: a platform for smart multicodal graphics search. In *Proceedings of the 22nd International Conference on 3D Web Technology*, page 17. ACM, 2017. (*Cited on page 133.*)

[198] Mohamed Sarwat. Interactive and scalable exploration of big spatial data–a data management perspective. In *2015 16th IEEE International Conference on Mobile Data Management*, volume 1, pages 263–270. IEEE, 2015. (*Cited on pages 133 and 134.*)

[199] Behrooz Omidvar-Tehrani, Plácido A Souza Neto, Felipe M Freire Pontes, and Francisco Bento. Geoguide: An interactive guidance approach for spatial data. In *2017 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pages 1112–1117. IEEE, 2017. (*Cited on pages 133 and 134.*)

[200] Corneliu Octavian Dumitru, Shiyong Cui, Gottfried Schwarz, and Mihai Datcu. Information content of very-high-resolution sar images: Semantics, geospatial context, and ontologies. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 8(4):1635–1650, 2014. (*Cited on pages 133 and 134.*)

[201] Jarek Rossignac. Interactive exploration of distributed 3d databases over the internet. In *Proceedings. Computer Graphics International (Cat. No. 98EX149)*, pages 324–335. IEEE, 1998. (*Cited on pages 133 and 134.*)

[202] Chao Chen, Daqing Zhang, Bin Guo, Xiaojuan Ma, Gang Pan, and Zhaohui Wu. Tripplanner: Personalized trip planning leveraging heterogeneous crowdsourced digital footprints. *IEEE Transactions on Intelligent Transportation Systems*, 16(3):1259–1273, 2014. (*Cited on pages 133 and 134.*)

[203] Slava Kisilevich, Florian Mansmann, Peter Bak, Daniel Keim, and Alexander Tchaikin. Where would you go on your next vacation? a framework for visual exploration of attractive places. In *2010 Second International Conference on Advanced Geographic Information Systems, Applications, and Services*, pages 21–26. IEEE, 2010. (*Cited on pages 133 and 134.*)

[204] Claudia Sousa Monteiro, Carlos Costa, André Pina, Maribel Y Santos, and Paulo Ferrão. An urban building database (ubd) supporting a smart city information system. *Energy and Buildings*, 158:244–260, 2018. (*Cited on pages 133 and 134.*)

[205] Yimin Yang, Wenting Lu, Jesse Domack, Tao Li, Shu-Ching Chen, Steven Luis, and Jainendra K Navlakha. Madis: A multimedia-aided disaster information integration system for emergency management. In *8th International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom)*, pages 233–241. IEEE, 2012. (*Cited on pages 133 and 134.*)

[206] Nan Cao, Chaoguang Lin, Qiuhan Zhu, Yu-Ru Lin, Xian Teng, and Xidao Wen. Voila: Visual anomaly detection and monitoring with streaming spatiotemporal data. *IEEE transactions on visualization and computer graphics*, 24(1):23–33, 2017. (*Cited on pages 133 and 134.*)

[207] Zeng-Wei Hong, Rui-Tang Huang, Kai-Yi Chin, Chia-Chi Yen, and Jim-Min Lin. An interactive agent system for supporting knowledge-based recommendation: a case study on an e-novel recommender system. In *Proceedings of the 4th International Conference on Uniquitous Information Management and Communication*, page 53. ACM, 2010. (*Cited on pages 133 and 134.*)

[208] Mahadev Satyanarayanan, Phillip B Gibbons, Lily Mummert, Padmanabhan Pillai, Pieter Simoens, and Rahul Sukthankar. Cloudlet-based just-in-time indexing of iot video. In *2017 Global Internet of Things Summit (GIoTS)*, pages 1–8. IEEE, 2017. (*Cited on pages 133 and 134.*)