NOVA SCHOOL OF SCIENCE & TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE

ALBERTO MIGUEL PIRES ALVES

M.Sc. in Computer Science and Engineering

A MODEL-DRIVEN APPROACH TO DATA VISUALIZATION

MASTER IN COMPUTER SCIENCE AND ENGINEERING

NOVA University Lisbon

April, 2024

NOVA SCHOOL OF SCIENCE & TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE

A MODEL-DRIVEN APPROACH TO DATA VISUALIZATION

ALBERTO MIGUEL PIRES ALVES

M.Sc. in Computer Science and Engineering

Adviser: João Carlos Gomes Moura Pires
Associate Professor, NOVA School of Science and Technology

Co-advisers: Maribel Yasmina Campos Alves Santos
Full Professor, Escola de Engenharia da Universidade do Minho

Ana León Palacio
Researcher, Universitat Politècnica de València

Examination Committee

Chair: João Baptista da Silva Araújo Júnior
Associate Professor, NOVA School of Science and Technology

Rapporteur: José Fabián Reyes Román
Researcher, Universitat Politècnica de València

Adviser: João Carlos Gomes Moura Pires
Associate Professor, NOVA School of Science and Technology

MASTER IN COMPUTER SCIENCE AND ENGINEERING

NOVA University Lisbon
April, 2024

A Model-Driven Approach to Data Visualization

This document was created with the (pdf/Xe/Lua)LaTeX processor and the NOVAthesis template (v7.0.2) [67].

To my family and closest friends.

Acknowledgements

My gratitude goes out to my advisor Prof. João Pires, and my co-advisors Maribel Santos and Ana León for the counseling, availability, and support throughout the development of my dissertation. I am extremely grateful for the opportunity given to me to work on this thesis and to consequently grow as an individual.

This was a truthfully challenging time, in all possible manners. It was something that I had not thought possible five years ago. From the first steps until this last leap, it was never a straight path.

Personally, I want to thank my close family for all the necessary help and support during the last 24 years. My mother, father, sister, auntie, and grandmother were instrumental in my upbringing and molding the person that I am today. They deserve the same amount of credit for the development of this thesis as myself, every single one of them.

Lastly, but equally important, I want to both thank and congratulate my friends, the newest and the oldest alike. I am extremely proud of them and wish for them to succeed in all possible fields and skip over every hurdle. A special word goes out to the group of friends that this university shaped (*M.S*), as they played a significant role in my life and provided so many fun times and support. Because of them, I strive to become a better person like themselves.

To all of them, thank you from the bottom of my heart.

”

“A própria luta para atingir os píncaros basta para encher o coração de um homem.”

— Albert Camus, O Mito de Sísifo

Abstract

The increasingly larger amount of data in the world creates a severe problem within data analysis, since experts, time, and computational power are limited resources. Therefore, systems must accommodate data analysis in a way that reduces the impact of all the three previous factors. Data visualization is one of the most popular techniques in data analysis and its impact is undeniable. However, it is a field of work that requires a lot of expertise that most individuals do not own. This is one of its main weaknesses.

This thesis focuses on addressing some of the challenges associated with visual literacy, user requirements, and domain-specific language translation. This will be achieved by developing the foundation for a data visualization approach based on the usage of conceptual models capable of abstracting domain concepts, taking into consideration domain questions, and recommending appropriate visualizations. To achieve this, our approach will be based on a conceptual modeling methodology, where the models express domains and these models are further expressed by a meta-model capable of characterizing them. A translation technique able to go from domain-specific natural languages to higher-level abstraction languages will be developed. The integration and expansion of a recommendation system, capable of leveraging user requirements when providing suggestions for visualizations, will also take place.

Strides toward the automation process of translating domain specifications, determining analytical tasks, and drawing visualizations through the usage of recommendation systems - in order to answer domain questions - are some of the contributions achieved throughout the development of this dissertation.

Keywords: Data Visualization, Recommendation Systems, Data Analysis, Visual Analytics, Conceptual Modeling

Resumo

O aumento exponencial de dados constitui um problema crescente inerente ao domínio da análise de dados. Os especialistas de dados, tempo e poder de computação são recursos basilares para esta disciplina. Contudo, estes mesmos recursos são escassos e têm variadas limitações. Tendo em conta este obstáculo, os sistemas modernos devem acomodar a análise de dados de forma a minimizar o impacto aplicado a estes elementos. A visualização de dados, uma das técnicas do campo da análise de dados, apesar de deter uma influência cada vez mais significativa, é ainda uma área de investigação/trabalho que necessita de indivíduos especializados. Este é um dos seus pontos fracos.

Um dos principais pontos de foco desta dissertação passa pela criação de soluções para desafios relacionados com a literacia visual, necessidades dos utilizadores e a tradução de linguagem natural associada a domínios específicos. Procuramos obter estas soluções através do desenvolvimento de fundamentos base para uma abordagem orientada a modelos capazes de abstrair variados domínios, tomar em consideração perguntas do utilizador e sugerir visualizações capazes de responder adequadamente às questões dos utilizadores. O desenvolvimento passará pela utilização de um meta-modelo capaz de caracterizar domínios, funcionalidades para traduzir e abstrair questões relativas a domínios e a utilização de um sistema de recomendações.

Durante o desenvolvimento desta dissertação, foram feitos avanços a nível do processo de tradução de linguagens do domínio, da criação de uma taxonomia de tarefas analíticas e da capacidade de oferecer assistência na obtenção de visualizações que permitam responder às questões do domínio.

Palavras-chave: Visualização de Dados, Sistemas de Recomendação, Análise de Dados, *Visual Analytics*, Modelação Conceptual

Contents

List of Figures

List of Tables

List of Listings

Acronyms

ASP Answer Set Programming *(pp. 27, 28, 66, 109)*

CDA Confirmatory Data Analysis *(p. 7)*
CIM Computation Independent Model *(p. 33)*
CM Conceptual Model *(pp. 34–37)*

DSL Domain-specific Language *(p. 34)*

EDA Exploratory Data Analysis *(p. 7)*

GPL General-purpose Language *(p. 34)*

IS Information System *(pp. 34, 35)*

LLM Large Language Model *(pp. 27, 53, 78, 79, 82, 85–89, 101, 102, 107, 108, 110)*

MDA Model-Driven Architecture *(pp. 32–34)*
MDD Model-Driven Development *(p. 32)*
MDE Model-Driven Environment *(p. 32)*
MOF Meta Object Facility *(p. 33)*

OCL Object Constraint Language *(p. 60)*
OMG Object Management Group *(pp. 32, 33)*

PIM Platform Independent Model *(p. 33)*
PSM Platform-specific Model *(p. 33)*

UML Unified Modeling Language *(pp. 33, 34, 60)*

1 Introduction

1.1 Context & Motivation

Why should we be interested in data analysis and data visualization? In a very straightforward manner, it is because data moves the world. Every day, organizations collect and process more data to better understand their specific domain. The widespread use of data mining and collecting tools increases the volume of available data, creating extensive data pools that cannot offer anything besides plain data. Although this data is precious and increasingly more valuable in the modern world, it usually lacks the ability to express anything other than itself. Therefore, achieving answers through raw data alone is an almost impossible task without thoroughly analyzing it first.

Data analysis, as a concept, is the critical evaluation of information in order to construct a deeper knowledge about the data at hand [26]. It is a broad field with numerous branches that divide themselves into differently focused domains with specific and concrete objectives. Data visualization, one of these fields, leverages the usage of visual representations of data as a supportive tool to help users build further knowledge about their data [91]. When designed correctly, the visual graphics are easily absorbed by the human information processing systems which allows for effectively interpreting information.

One of the main motivations for automating data visualization is to fill the gap between users. Specifically, because users are very different. Some users might be well-versed in visualization topics, while others do not know the best course of action to approach data and/or data visualization, due to their limited knowledge [120].

To push through this challenge, experts rely on establishing tasks, in order to achieve the user's goals, by guiding the user into translating high-level tasks that result from domain-specific problems into lower-level ones that can be turned into recurring tasks that improve the design and evaluation of visualizations [106]. For example, person *A* wants to know the price gap of item *B* in multiple supermarkets. In this example, "wanting to know the gap" could mean something like **Comparing** or perhaps **Correlating** the price in various stores, which is a commonly performed analytical task. Although simple, this kind of example conveys an idea of how everyday and analytical speech differs.

This dissertation is framed within a research initiative surrounding the development of a model-driven approach for data analytics. Such an initiative combines knowledge from three different institutions, in an effort to achieve progress breakthroughs regarding our hypothesis. This team is comprised of five elements:

- **MSc Students** - Alves A. (NOVA School of Science and Technology) and Almeida A. (Escola de Engenharia da Universidade do Minho).
- **Professors** - Pires J. (Advisor, NOVA School of Science and Technology) and Santos M. (Co-advisor, Escola de Engenharia da Universidade do Minho)
- **Researchers** - León A. (Co-advisor, Universitat Politècnica de València)

Figure 1.1 displays the point of reference of our approach's concept, in which this dissertation is inserted, its underlying logical steps, and foundational reasoning. This diagram follows an informal notation that allows us to emphasize the components, their connections, and the resulting interactions.

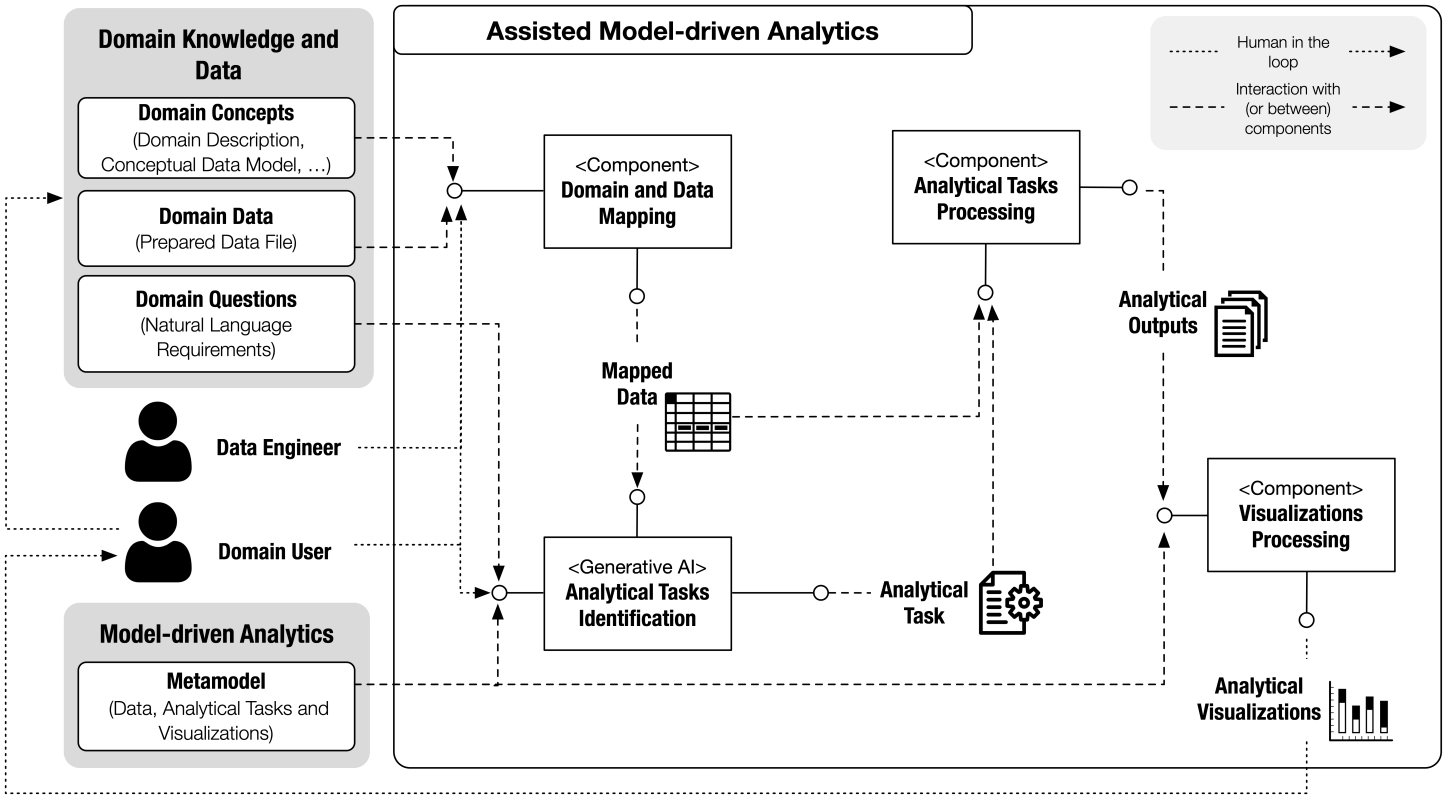

Figure 1.1: Assisted model-driven analytics approach, credits go to Pires J. (main advisor), Santos M. (co-advisor), and León A. (co-advisor).

Our iterative model-driven approach is composed of four main components:

- **Domain Knowledge and Data** - The domain's conceptual model describes the concepts and relationships of said domain. This component is composed of **Domain Concepts**, **Domain Data**, and **Domain Questions**.

- **Model-driven Analytics** - Denotes the meta-model, our foundational conceptual model, which defines how data, analytical tasks, and visual choices shall be interpreted.
- **Humans in the loop** - Both the **Data Engineer**, the expert that aids the user for the duration of this process, and **Domain User**, responsible for establishing the domain question, are crucial agents in the iterative loop of this approach.
- **Assisted Model-driven Analytics** - Characterized by four main steps, working in a linked way, that state the conceptualization of data, the mapping of different data, the identification of analytical tasks, and finally the visualization itself. **Domain and Data Mapping**, is the first task and it will map the data relative to the domain, it confirms the relationship between the concepts and the existing data. It outputs mapped data that is ready to be consumed; **Analytical Task Identification**, uses the mapped data from the previous tasks and translates the domain questions into analytical tasks, both the domain user and the data engineer (data expert) must be present in the process of this step. Results in a list of analytical tasks that compose and characterize the initial domain question; **Analytical Tasks Processing**, evaluates the outputs of the aforementioned steps (Domain and Data Mapping and Analytical Task Identification) in the form of mapped data and associated analytical tasks. In this step, the designated tasks serve as the guidelines by which the mapped data will be transformed. The respective outputs result in the analytical outputs that are in concordance with the meta-model (conceptual model); **Visualization Processing**, with the analytical outputs in conjunction with the meta-model information it is possible to design adequate analytical visualizations, by resorting to a recommendation system.

1.2 Hypothesis & Objectives

Our hypothesis stems from wanting to unfold inherent challenges (e.g. difficulty in the initial analysis of domains) in the data visualization field through the usage of conceptual models as the driving basis of our research. In data visualization, user requirements constitute a number of possible problems. Being capable of leveraging both the domain abstract concepts and the user's domain questions is not an easy task. It can be hard to grasp user requirements because they usually closely resemble the natural language idiom with a domain-specific vocabulary. These requirements have to be translated into knowledge in order to be further translated into a more generic level, to analytically distinguish them - For another example, a scientist wants to know if the results of treating tissue with substance A match up with tissue treated without substance A. The domain vocabulary **Match up** translates into a **Comparative** analytical task [80].

Generally, this work tends to be strenuous and it necessitates the cooperation of a capable data expert, especially when talking about users with non-existent or very limited

visualization literacy. Factoring this side-by-side with the number of tools and research that still lack the prospect of mapping these requirements, it becomes clear that this hypothesis faces challenges that are relevant in today's state-of-the-art.

- The first challenge is related to extending a conceptual model that works as the basis of the project. The soundness of this model must be paramount since it works as a foundation for all other advancements made throughout this thesis.

- Creating broad enough, but simultaneously complete, tasks capable of addressing a multitude of possible domain questions and commonly seen data analysis patterns. This will allow for the development of a tailor-made taxonomy that is thought from scratch to build a symbiotic relationship with the conceptual model.

- Translating and mapping domains, we are simultaneously trying to model different types of data to connect them to the respective domain concepts, also expressed in the model, with the current data. This reveals two potential problems, related to the formulation of analytic requirements based on the domain-specific language (since they might be inaccurate) and how they must be translated adequately to analytical task(s).

- The final challenge is the correct execution and rendering of the data visualizations, in an efficient manner, after all necessary tasks were performed. Our project team also expected to integrate a recommendation system that, based on the previous challenge, would be capable of guiding the user to the most appropriate visualizations possible. This might be problematic because recommendation systems take a lot of information into account, especially since this approach will add more details to that load.

We [the Team] believe that our advancements, in unison with our initial conceptualization of the model-driven analytic hypothesis, would become an important piece of architectural structure from which to create a proof of concept, that would be advantageous knowledge for future work. These objectives and challenges would go hand-in-hand during the development of this dissertation.

1.3 Approach & Contributions

The primary purpose of this dissertation is twofold. Establishing a proof of concept investigation capable of providing a thoughtful path from which to build our initial approach, and simultaneously laying a solid foundation for future work to be developed beyond the scope of this thesis.

These objectives were set to be achieved by expanding upon the existing state-of-the-art research, particularly by combining features not commonly found within data visualization literature. On top of that, another key focus was the prospect of having an

ontologically sound and efficient basis, suitable to be explored onward by other researchers, as a control benchmark, or as a standard to improve upon.

In terms of contributions for work done during this thesis, it is available:

- **(1)** Extension and completion of the **Meta-Model**, adding and extending components that will describe how the data is mapped to the visualization and continuously make changes that follow the evolution of the project.
- **(2)** Constructing a fully functioning **Taxonomy** that was custom-built to integrate the meta-model. The executed research, comparative analysis, vital thought process, tests conducted, and the finalized proposal, were some steps taken toward achieving our taxonomy.
- **(3)** Choosing and expanding a **Recommendation System** capable of handling the possibility of automation, with partial or full integration of our taxonomy within the model itself, and able to design adequate visualizations that can be considered worthwhile to answer the domain questions.
- **(4)** Testing the possibility of using **LLM Technologies** to translate domain questions directly into the analytical tasks comprising the proposed taxonomy.

In terms of contributions within the team: **(1)** was an effort mainly conducted by Almeida A. where I helped with necessary changes to the model in the analytical and data modules, and the addition of the visualization module; **(2)** was a team effort where myself [Alves A.] and Almeida A. surveyed different taxonomies and did a deep analysis regarding the existing tasks to create our proposal, which was reviewed and re-evaluated in cooperation with the remaining of the team (Pires J., Santos M., and León A.); **(3)** was done exclusively by myself, although the decision of which system was made with the approval of the advisors; **(4)** was mostly executed by myself, with the help of the advisors in overseeing and adjusting some of the domain questions.

This dissertation's GitHub is publicly available and can be visited on the annex II.

1.4 Document Organization

The dissertation will be organized in the following manner:

- **Chapter 1:** Introductory chapter addresses the theme, context, and primary objectives of this dissertation.
- **Chapter 2:** Background chapter is where the foundational concepts of this dissertation will be meticulously explained. Includes the analysis of the current state-of-the-art in the existing literature.
- **Chapter 3:** Proposed Taxonomy chapter, lays a foundation based on research, analysis, and posterior evaluation that construes and illustrates a taxonomy and its utilization.

- **Chapter 4:** Modeling focused chapter that will set forth the developments made on the conceptual model, the proposed taxonomy, and finally the data repository.
- **Chapter 5:** The Recommendation System chapter delves into the chosen recommendation system, the expansion of features, the integration into this dissertation, and finally its usage.
- **Chapter 6:** Evaluation chapter analyzes the conducted research and the focused tests that were led in multiple instances. The main objective is to delineate correctly the evaluation of these results and how they compare.
- **Chapter 7:** In the last chapter, Conclusions, there will be an overview of the research and advancements made throughout this dissertation, together with proposals for future work development and some final observations.

2

Background & Related Work

The following chapter will lay crucial background knowledge from which this dissertation will use as a foundation. It will encompass core concepts about Big Data, Data Analysis, Data Visualization, and Modeling by gathering insight regarding recent and relevant research on how these concepts evolved, their standards, and the current paradigms in the literature. Ultimately, with the objective of linking these concepts to the work that was performed during this dissertation.

2.1 Big Data Analytics & Data Analysis

Big Data conveys the idea that within a certain system, the data outgrew the storage that would be typically expected. On the other hand, **Analytics** is a branch of Data analysis that extracts new information from old data with the support of multiple tools and techniques. [82]

The junction of both concepts creates, what is commonly known as, **Big Data Analytics**, which is defined by the execution of advanced analytic techniques on big data pools with the main objective being the extraction of valuable information for decision-making purposes.

[Advanced] Analytics comprises numerous techniques and tools in order to be correctly employed, like predictive analytics, data mining, **Data Analysis**, **Data Visualization**, artificial intelligence, language processing, and complex databases.

Data Analysis, a subset of Analytics, is a process of data manipulation with the intent of exploring data and extracting information using various available techniques [26]. It offers two main branches: Confirmatory and Exploratory.

Confirmatory Data Analysis (CDA), is an approach within data analysis where existing models or hypotheses are proven or refuted by statistical approaches. This methodology targets a test-based numerical procedure to the detriment of visualization, through statistical testing [39].

Exploratory Data Analysis (EDA), focuses on acquiring knowledge by means of exploration as a result of not having a solid hypothesis to prove. The shift from focusing

on gaining information from data instead of confirming a specific objective explains why visual methods, especially **Data Visualization**, are such a fundamental technique. Precisely because, in effective visualizations, it is easy and clear to find relationships that can hold beneficial information and guide us on the right path towards building a model or hypothesis, instead of forming them *a priori* [121].

Big Data Analytics works because of the reciprocal relation between concepts: Big data is capable of providing samples, which enhances analytic results, and there is a lot to gain from unstructured and unfiltered data, especially in large volumes of data [97].

Although Big Data is often associated with the volume of data, this would be a very rudimentary way to look at this field. The defacto characteristics of Big Data [37], are:

- **Volume -** Primary attribute of Big Data. Usually measured in the size of the data, number of records, transactions, tables, files, and time.
- **Variety -** Evolution of data extraction and data mining created a wide range of sources from which data can originate, this creates variety. Data within the same dataset, or that fits into a determined domain, can have vastly different types.
- **Velocity -** Frequency in exchange of data, either as input or output.
- **Veracity -** Represents the unreliability inherent to data, which is highly volatile and therefore is also susceptible to being unreliable at times.

Regarding Big Data's potential, it is critical to effectively process data to achieve goals. Understanding the process of outputting reasoning from Big Data can be thought of as a series of processes, divided into two stages: data management and analytics [37]. Data management is responsible for mining data and pre-processing it for analysis. Analytics is responsible for modeling and extracting information from data, as seen in Figure 2.1.

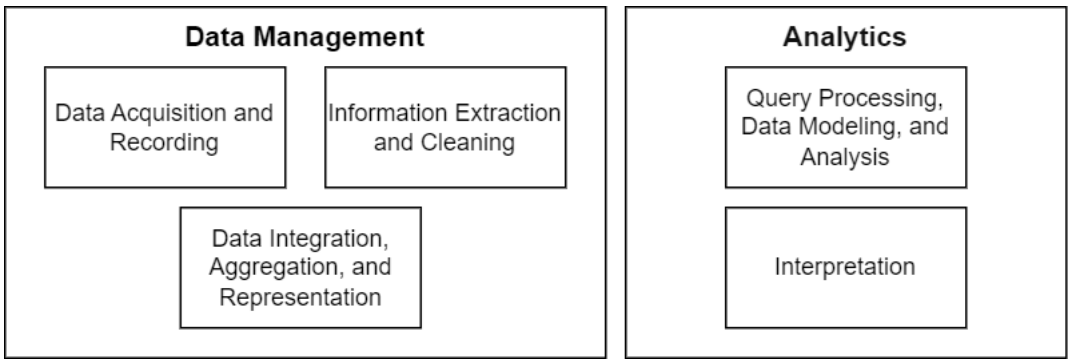

Figure 2.1: Big Data processes pipeline, adapted from A. Gandomi and M. Haider [37] and Agrawal et al. [3].

The pipeline [3] can be described as:

- **Data Acquisition and Recording -** Acquisition of data is becoming easier, multiple tools and techniques are capable of retaining large quantities of data. Having

substantial amounts of data comes with the problem that certain data entrances have no real value, therefore they must be filtered. Categorizing data in a way that makes it easily contained and accessible, in case of being analyzed, is an elaborate task.

- **Information Extraction and Cleaning -** After being stored and filtered, data must be formatted by keeping the relevant information from the original data. It is extremely system-dependent since different domains will have different types of data, and objectives.

- **Data Integration, Aggregation, and Representation -** Besides basic procedures like locating, identifying, and understanding the data, data analysis when seen through the big data lens requires automation for its correct and effective functioning. This creates a new challenge in which we need to define the data in a computer-friendly manner.

- **Query Processing, Data Modeling, and Analysis -** Big Data has a special view since the amount of data, even when problematic, is statistically more likely to have information about patterns and relationships than other small data pools. Data mining can help improve data quality and provide querying functions.

- **Interpretation -** Results are only as good as the user that interprets them. These interpretations involve much deliberation and examination. Communication and how data is presented to the user is also significantly important.

Above all, Big Data needs context. Since most analytical questions will be born from a data-driven perspective, this requires users to acquire another layer of higher understanding of the data and respective domains. As good data visualization analysis tools become increasingly relevant in displaying information to users, in a simple and straightforward manner, analysts will have to become familiarized with data visualization in order to present results that assist the ideas behind the interpretation.

2.2 Data Visualization

Often considered an important resource in data analysis, visualization represents a logical point of connection between data objects and the Human visual processing power [36]. The ability to perceive visual elements from dimensional spaces while being capable of establishing connective structures to harvest knowledge of implicit information accurately [96], explains why visualization is such a multidisciplinary field. That withdraws interest from other fields that necessitate or might benefit from the usage of visualizations to get a proper insight into available data. From the research conducted, the visualization field itself has two main areas: **Scientific**, where the data is directly associated with real-world experiences/experiments that tend to use intrinsically heavy spatial and temporal data

as distinctive elements [34]; and **Information**, where the visualizations aim to use data that is representative of abstract concepts that can be associated with some real-world equivalents. In this dissertation, our approach is set to be most closely related to the information branch since the abstraction of existing concepts will play a pivotal role.

Although powerful, visual displays should be thoughtfully curated and deeply concerned with the way they express data. Due to its potential subjectivity and interpretative nature, the possibility of creating ambiguity, optical illusions, and assumptions, from the observer's perspective, is a fairly common problem [36]. For example, two graphs with similar data expressed on them can create broadly different end results. Hence why there is a need to label, detail, and highlight every major part of visualizations.

Designing good visualizations is not trivial. For instance, Figure 2.2 creates an illusion that makes the values of the circles, and bars, seem far apart. In reality, this happens because the user that created this visualization had a non-zero axis and did not apply any axis transformations. A hypothetical case about productivity between worker A and B could make the owner of a company believe that worker B was a much better employee than worker A, even though the difference is only marginal.

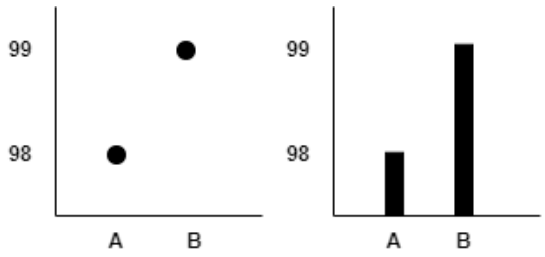

Figure 2.2: Example of a common illusion in visualization, taken from S. Franconeri et al. [36].

The core objective of data visualization is the conceptualization of new, relevant information, and the acquisition of deeper levels of knowledge about the data. Consequently, improving the knowledge about the domain as well. Achieving this objective can be extremely difficult since the analysts responsible for showing the data must create good enough visualizations that allow viewers to attain the same conclusion as them [5]. Thus, visualization has three main supportive mechanisms [5, 106, 26], in order to reach such an objective: **Exploratory Analysis** - Users want to discover something based on the data they hold. Through the usage of analytical processes, it is possible to find the existence of patterns, trends, and relations; **Confirmatory Analysis** - Using knowledge about the domain, the user creates a hypothesis and the objective of this analysis is to discover if the visualization is capable of confirming or denying the base hypothesis; **Presentation** - Draws the visual component that highlights the relationships, structure, behavior, and other intrinsic characteristics of the data involved in the problem.

2.2.1 Data Abstraction

A significant part of visualization works within a highly hypothetical thought process. Abstraction in data visualization leans towards the conceptualization of abstractions utilizing/combining fields that can provide a bigger basis for visualizations to strive in.

There are numerous ways to implement data visualization principles/concepts to already established systems, if those systems are thought of as being capable of taking advantage of data visualization to produce, achieve, and acknowledge meaningful results, for example: Designing visualizations as knowledge generation tools or knowledge-enhancing systems, where the visualization is either interconnected with a direct or indirect abstraction of the human interaction system, creating the opportunity for the user to gain knowledge of a specific proposition, by virtue of the visualization [98]; Quantifying the knowledge gained from visualizations, in a model, through mathematical functions/algorithms that try to simulate and abstract upon the real-world setting [128]; Integrating visualization within probability theory principles, in order to achieve reasoning by a visual representation of the agent interaction with the environment [79]; Through the spectrum of semiotics, of how humans interact and perceive information originating from signs, by using category theory.

2.2.2 Visualization Pipeline

Independent of the approach, the steps necessary to obtain visualizations typically follow the same general process. Commonly known as "Visualization Pipeline" or "InfoVis Pipeline", it has multiple models and proposals - for example, Card et al. [22] that is widely accepted as a reference model, Chi [23], and Santos & Brodlie [101] - that although different in terminologies and steps, have the same logical roots.

The main motif behind pipeline research is to investigate further the process by which systems handle data and map it into visual elements. These models stratify the basis on which data visualization should theoretically work, starting with the data processing and ending with the visualization itself.

In Figure 2.3, there is a pipeline initiated with the intake of raw data. This data will then undergo data transformation and manipulation techniques that are deemed to be fit to build the visualization. Afterward, when the data is already structured, homogeneous, and filtered is going to be mapped into geometric primitives, with their respective visual channels. Finally, this information is grouped together and ready to be rendered into the respective visualization space. These set of steps tend to be common in data visualization system's pipelines.

There are also alternatives to this methodology of the visualization design process, Demiralp et al. [31] theoretical model inspired by visual embedding functions that are capable of preserving structures in the data into the embedded space. The process of mapping works through the execution of Cartesian products of all visual elements and then by evaluating the estimated perceptual distance embedded matrices. As stated by

the authors, this build of visualizations based in tensors has different challenges and requires specific tools to handle it. Another alternative theoretical model, by Kindlmann & Scheidegger [57], proposes the application of pure algebraic principles, like invariance, in order to reason about the mapping of data.

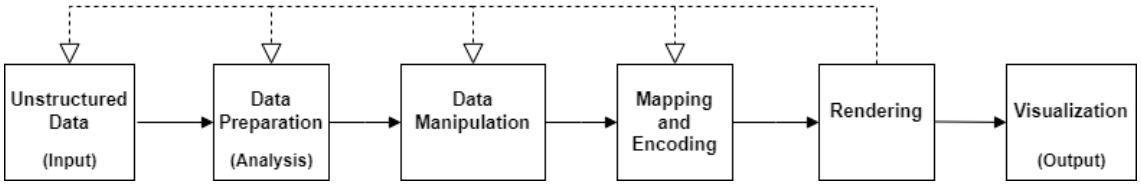

Figure 2.3: Data Visualization Pipeline, taken from Qin et al. [91].

2.2.3 Effective Visualizations

As a tool, visualization offers excellent communication of information in a straightforward approach, and this is one of its main advantages. Therefore, visualizations must capitalize on such factors and be both efficient and accurate. In **Effective Visualizations** it should be easy and effortless to understand the intended interpretation of a unique visualization [36].

Visualizations rely heavily on the following concepts [110]:

- **Tasks** - Influence the demands within a certain visualization.
- **Users** - Interact with data and visualizations. Humans can outperform computers at finding patterns or underlying information, in some cases. Ultimately, users are also responsible for deeming if visualizations are correctly conveying information.
- **Data** - Data types and structure are two catalysts behind design choices (visual elements, organization, chart type, etc.) in visualization spaces.

Analyzing the design choices of a visualization can be tricky without a base model due to the multiplicity of possibilities when accounting for tasks, data, and goals. According to Münzner [80], visualization design has four nested levels, as it can be seen in 2.4: **Domain Situation** - describes the domain, its vocabulary, workflow of data, and the interests of the user. Fundamentally, it serves to understand the goals of the user; **Data/Task abstraction** - abstracts upon data and tasks, offering higher-level versions of the original domain-dependant context. It helps determine analytical tasks and gives the opportunity to create new data from the existing one, if necessary; **Visual encoding/Interaction idiom** - at this level is where the main design decisions relative to the visual elements are taken, both the creation and the manipulation of the visualization happens at this stage; **Algorithm** - last level gathers all previous decisions and sends this information to a tool capable of automatically handling the drawing of the visualization.

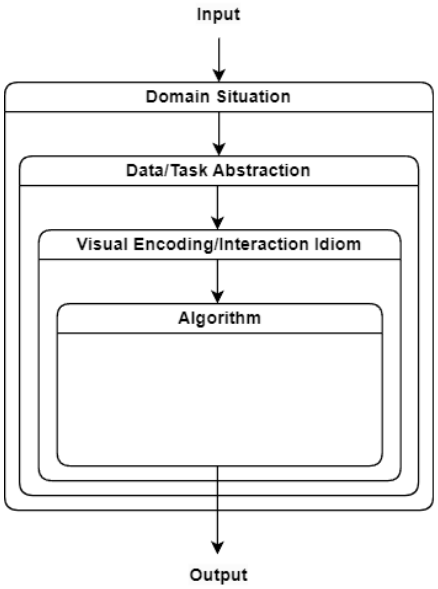

Figure 2.4: Nested framework of analysis, adapted from T. Münzner [80].

Knowledge and details are important parts of choosing and pondering about what is the most appropriate visualization. Tasks are usually very detail-dependent because they originate from users and knowledge about the data.

The more details we are able to determine within a certain task – for example, comparing values, statistical estimations, or finding trends - the more we can be sure of the best possible visualization for that task. This happens because users, who communicate their goals, tend to do so in a language that is highly related to the domain of the problem, which can hide the true analytical goal of the task. Due to the lack of abstraction, these languages often are unclear in determining their goals and must be translated into higher-level languages. Users are responsible for creating the requirements that personalize each specific task. The structure of data and domain are characteristics that help determine what type of visualization is more effective.

Common issues that hinder the validity of visualization stem from perceptual errors, like optical illusions and misperceptions. Taking errors into consideration is crucial when thinking about accurate visualizations - for example, an excessive amount of elements, inadequate mapping ratios, illusory contrasts, incorrect visual elements, and imprecise graphs are some of the problems that an analyst wants to avoid in his visualization development [36].

2.3 Visual Components

In **Data Visualization**, the visual components are an essential part of information display. **Visual Components** are necessary in order to express data elements in display, in a way in which the human visual system can perceive them. Visual components are typically associated with two main concepts: marks and channels [80].

Marks are simple geometric primitives that represent items or links between items in a visualization space. In other words, they represent a visual mapping of a data element into a visualization. These marks are a very basic component that can be enhanced and specialized into new roles through the usage of visual channels.

Visual channels control the way in which a mark is displayed in space, they connect through mapping chosen visual cues into specific marks or groups of marks. This property of marks, which goes by an extensive number of names, is responsible for assigning multiple types of visual cues like position, color, value, text, shape, angle, size, and texture [14]. These channels can be encoded into marks, even redundantly if necessary, in order to express more information in our visualization and to enrich data. Depending on the objective and data type, different variables have different characteristics - Selective, Associative, Quantitative, and Order - relative to their perceptual handling that make them more, or less, suitable [70].

Since visual components are of exceptional importance, it is expected for them to be an integral part of research. Fundamentals about perception, color theory, semiotics, effectiveness and accuracy, and cognitive psychology are necessary to understand the underlying value of each component. As one of the core concepts enabling us to display data efficiently, it is rather important to study and acknowledge how they work and influence the observer's conclusions.

Bertin, a central figure in visualization theory, published his study about semiology [14], the science responsible for studying the communication of verbal and non-verbal signs. Such a study lays down the principle of mapping data to visual elements in a way that the human visual system was efficient at perceiving, with base concepts like marks, positional planes, and retinal variables (visual variables). He also defended that visual variables conveyed particular characteristics with distinct expressive power. These characteristics were: **Selective** - finds that a single visual symbol can stand out from others; **Associative** - confirms that a series of visual symbols can be grouped together based on similarity; **Quantitative** - visual symbols are capable of presenting absolute values; **Ordered** - defines that a series of visual symbols can be ordered into a sequence.

Marks	Points, lines, and areas (Geometrical Primitives)
Positional	Two planar dimensions
Retinal	Position, size, shape, value, hue, orientation, and texture

Table 2.1: Graphical Grammar proposed by Bertin [14].

Following this definition, Mackinlay extended on Bertin's fundamentals and assembled a new graphical basis with more computer-defining characteristics and vocabulary [70]. This is crucial to address since Bertin focused his study mostly on paper and physical media. Therefore, Mackinlay's is a more grounded structure to our visualization standards when compared to Bertin's. The study sets a basis for primitive graphical languages and

ranked the visual variables by their effectiveness when encoding different types of data, which was influenced by Cleveland & McGill's perceptual ranking [24].

Finally, Münzner's graphical language and effectiveness ranking [80], which is considered a literature standard and point of reference. Expressiveness and effectiveness appear as two crucial subjects. The first dictates that visual channels should only express the required information, while the latter relates to the accuracy of spotting a visual element within a visualization. Visual channels, as the author designates them, besides having new additions are also divided into two categories: **Identity channels** (Metathetic), represent categorical data - like color or shape - and communicate information about what a certain item is or its location; **Magnitude channels** (Prothetic), encode ordered data - like position, length, or tilt - and communicate how much of an item exists and its symbolic weight.

Figure 2.5 shows Münzner's impactful ranking, alongside it came a significant amount of insight to the landscape of data visualization that consequently altered how these channels are used and evaluated in state-of-the-art research. This type of ranking and classification is tightly associated with psychological research that studies human visual processing principles and different visual phenomena, like **pre-attentive properties**.

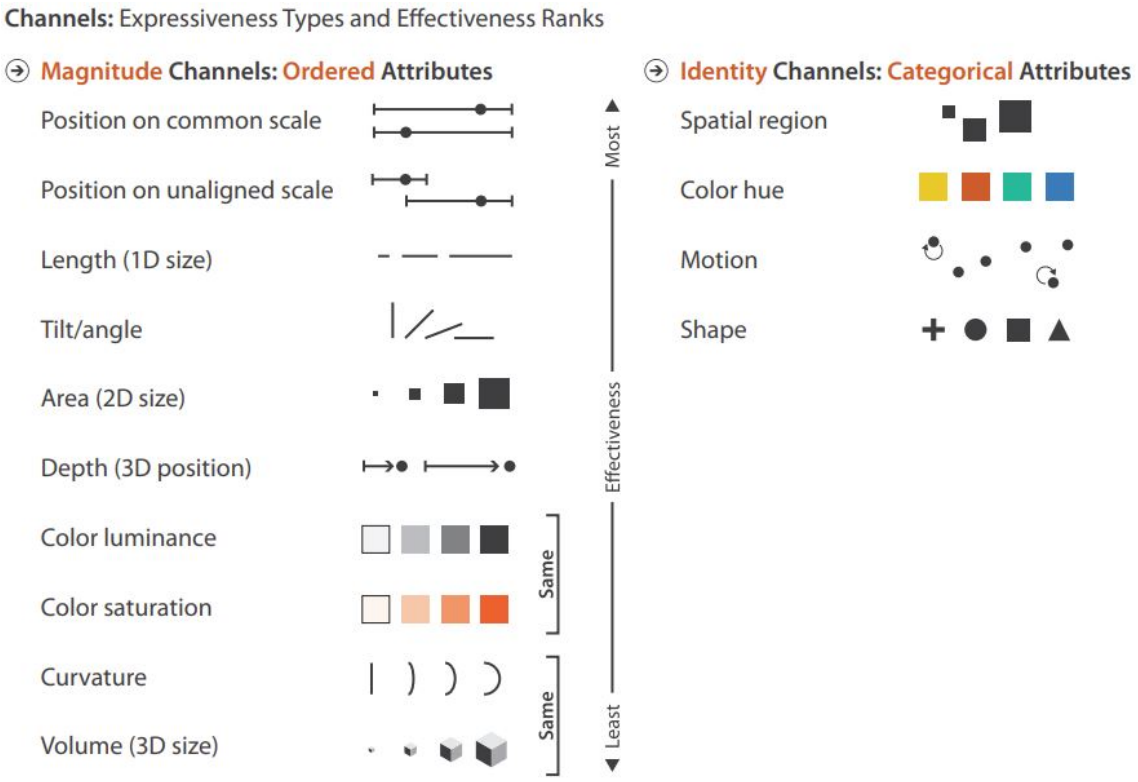

Figure 2.5: Channels categorized by their expressiveness and ranked by their effectiveness, taken from T. Münzner [80].

2.3.1 Perception

Perception is a human characteristic that uses the human senses in the interest of processing information. It is an always-occurring process of creating pieces of knowledge, that can be individually consumed, with the objective of assimilating them together into a whole. Hence, perception functions in a multi-process manner that establishes relations between elements, creating a perceptual structure of information that guides the understanding of the fundamental elements of an environment. In visualizations, mapping data to visual elements is an example of such. In effective visualizations, the connection between data is structurally identical to the perceived connection between visual elements [28], this should be glaring to the user by virtue of good design choices.

In the visualization spectrum, the approach is almost always solely made with sight as the primary sense of focus. Sight secures and retrieves information effectively, it is the main component of our sensory system at doing so due to its parallel processing capacity that lets the viewer grasp information from multiple points of reference simultaneously.

During the parallel processing phase, the visual system studies the data by highlighting visual information such as color, shape, size, position, texture, etc... The process by which visual elements are accurately and quickly detected is called pre-attentive since the finding of these visual cues happens before our attention is focused on the "bigger picture". After that, the initiation of the pattern-processing phase matches the scanning of the visual space in search of patterns and applying object recognition principles based on the information extracted in the previous phase. And finally, the sequential processing phase where the aggregation of knowledge associated with the resulting analyzed image facilitates the viewer's goals. This three-phased system is a simplified model of the information processing system that is necessary to work in order to gather results from visualizations [125].

2.4 Analytical Tasks & Taxonomies

During the designing of visualizations, one of the core notions is the utilization of data to answer users' questions that are associated with a specific domain, carried out by the available data. Although relevant, users' requirements are usually unclear, since users tend to express them in a language that does not reflect a task standardized name. A common scenario is the utilization of a particular domain language that needs to be analyzed in the pursuance of a more abstract meaning that precisely conveys what the user wishes to be answered. This translation leverages knowledge and enables critical analysis of affinities and differences between tasks.

Analytical Tasks specify a goal and tend to be split into high-level and low-level, although some research also includes a middle level which is reserved for searching operation, often based on the attributes and/or location of an element [80].

Tasks associate themselves with data and how data is displayed in the visualization

space. The reciprocal relationship between these two factors creates a feedback loop in which tasks can be influenced by data types and structures, or data might be altered (transformed, aggregated, derived, etc.) with the intent of accommodating a specific task [80].

By its formal definition, **Taxonomy** is the classification of critical concepts related to a specific domain, typically characterized in a hierarchical structure. It can aggregate akin concepts and define their existing relationships [124]. Fundamentally, taxonomies can be distinguished into low-level, as previously stated, and high-level. These are comprised of tasks, namely decision-making (although it does not usually appear in literature), exploratory analysis, causal effect, knowledge discovery, presentation, problem-solving, and sense-making.

When researching **Data Visualization** it is common to witness different approaches and, as a consequence, a lack of structural standard of implementation. Judging the importance of data visualization in a wide spectrum of fields it becomes apparent that such a standard taxonomy would have to take into consideration the multitude of scientific fields and their respective connections with the core concepts that constitute data visualization. Therefore, the more intricate a field is, the harder it becomes to translate into a taxonomy.

Typically, taxonomies within **Data Visualization** tend to define a unique field while shedding light on important concepts regarding visualization like data, mappings, encoding, and users.

2.4.1 Low-Level Taxonomies

Low-level tasks are thought of as simpler actions that can be performed by looking at the visualization space without the need to deeply analyze the bigger picture information. They typically reflect the same degree of difficulty as locating, identifying, or comparing particular item(s) [106] - comparing the price of item A in store B and C is a low-level task.

In their proposed taxonomy, Zhou and Feiner [137] suggest that visual tasks can be thought of as abstracted visual techniques that can be reached by the usage of low-level analysis techniques. In their proposal, the taxonomy exhibits multiple low-level tasks that serve the purpose of achieving high-level abstractions. The tasks at hand are closely related to the user's goals and what they expect to achieve. Commonly found analytical tasks such as **Associate**, **Categorize**, **Compare**, **Correlate**, **Distinguish**, **Identify**, **Locate**, and **Rank** are available in this work because they appear often within user questions, for example wanting to know if there is a correlation between event A and event B. These tasks can then be used, usually in groups, to attain information relating to higher-level techniques, such as **Elaborate**, **Summarize**, **Explore** and **Compute**.

This proposed taxonomy is heavily influenced by both the necessity of mapping user tasks to object types and by one of the first studies, in the low-level taxonomy field. In which Wehrend and Lewis [127] classified their taxonomy accordingly with eleven tasks: **Associate**, **Categorize**, **Cluster**, **Compare**, **Correlate**, **Distinguish**, **Distribute**, **Identify**,

Locate, and **Rank**.

2.4.2 High-Level Taxonomies

Relatively to high-level taxonomies, the appearance of taxonomies capable of using visualizations in a flexible manner within a design space become increasingly popular. **High-level tasks are considered to be complex actions that rely on finding patterns not explicitly expressed in the data**. Therefore, they have to be rationalized, such as frequencies, correlations, and distributions - for example, searching for a possible pattern between the manufactured numbers of item *A*, *B*, *C*, and *D* in different factories, is a high-level task.

Therefore, Heer and Shneidermann [47] developed a new taxonomy of interactive dynamics - setting its primary goal as being a supporting framework for comprehension and reasoning about analysis tools - composed of three categories:

- **Data & View specification** - Views are an essential part of data analysis, notably in the presence of substantial amounts of data, because of the possibility to create areas of focus that eliminate irrelevant data for the current context. Corresponding techniques that contribute to specification, as stated by the authors, are **Visualizing** available data, **Filtering** data in order to construct a smaller and more significant dataset, **Sorting** data accordingly to express trends and patterns, and **Deriving** objective information from existing data, either by applying transformations, creating new attributes, or removing data.

- **View Manipulation** - Succeeding the creation of Views, it becomes necessary to perform actions that enable sharpening the efficiency and accuracy of visualization. **Selecting** data object(s) to execute actions upon, **Navigating** the visualization space concerning the examination of specific data, **Coordinating** multiple visualizations with multiple views that require **Organization** of their respective dashboards and order of data narrative.

- **Analysis Processing & Provenance** - Analysis comes down to the process of exploring data and attaining knowledge about the aforementioned data. To enable this analysis process it is required to **Record** actions previously executed by the user, **Annotate** observations, information that the user deems necessary, or points of interest within the display; **Share** and communicate results, considerations, conclusions, etc., with other users, and **Guiding** users through the process of analysis.

2.4.3 Multi-Level Typology

Typologies are typically considered to be multidimensional and conceptual, though taxonomies tend to characterize empirical objects of study or features [12].

Brehmer and Münzner's research proposed the concept of a multi-level typology [19] that would be able to serve as a cognitive bridge between abstract taxonomies, low-level and high-level, through intricate means of characterizing complex tasks as an arrangement of medium and simple tasks. The main solution is the need to disambiguate the task at hand, and this is achieved by answering three leading questions: **Why** is the user interested in executing a task; **How** is the task going to be executed; **What** are the task's data and targets. The authors state that the importance of these questions is their complementary properties, since low-level taxonomies provide, mainly, answers to the **How** question. Whereas, high-level questions present solutions to the **Why** question. Together with the **What** question this typology creates a succinct and straightforward way to designate different tasks.

This type of approach is particularly interesting when establishing relations/connections between tasks due to their ability to reflect transactions from the domain problem to abstract tasks. Structurally, although they behave similarly, they have vastly different sub-categorizations:

- **Why** - describes the need behind the execution of a task. It is divided into three levels, high (Consume & Produce), mid (Search), and low (Query). **Consume** refers to the information that will be consumed by the visualization itself. This can be manifested in presenting the visualization to communicate the information. **Discover** focuses on the creation and proof, or refute, of a hypothesis based on existing information. **Enjoy** which simply reflects a more informal perspective that is not motivated by the need to verify any particular conclusion.

  - **Search**: is an extension of *Consume* in which the objective is to find a data element, or multiple (group/cluster), that are important to the exploration of data. Within this level, there are four possible tasks, depending on the target and its location (if they are either known or unknown). Searching for specific targets encompasses the *Lookup*, if the user knows the target's location, and *Locate* if the target's location is unknown; however, if the target is undisclosed then the possible tasks are *Browse*, if the target's location is known, and *Explore* if the location is unspecified.
  - **Query**: Following the discovery of the target(s), it is possible to engage in low-level tasks and extract information from relevant targets using tasks such as *Identify*, finding and identifying an item, *Compare*, directly compares a group of items, and *Summarize* that offers an overview of the data.

- **How** - It comes as a conclusion of the previous "What?" question, it describes the methodology behind the execution of the task and its interaction with the visualization. Categorized into *Encode*, responsible for mapping data objects into visual elements, *Manipulate*, applies methods that interact and influence visual

elements within the visualization space by transforming data - Select, Navigate, Arrange, Change, Filter, Aggregate are the available methods. *Introduce* adds elements to the visualization internal and external space by *Annotating*, textually or graphically, elements; *Importing* adds new data to the visualization; *Derive* creates new data from available data; *Record* keeps a state or artifacts in a history-type menu.

- **What** - Establishes what are the appropriate inputs and outputs of the visualization. Information relative to *Data Types*, from the possible items, attributes, links, positions, and grids; *Attributes Types*, represent the inner data type of each attribute within items, such as categorical and ordered; *Dataset Structure*, defines the overall structure of the available dataset (i.e. table, network, fields (continuous), multidimensional table, tree, or geometry).

In an attempt to polish this foundation, a further enriched multi-level typology, closely resembling the previous one, re-utilizes the three questions mnemonic **Why**, **How**, and **What**, that was proposed by Münzner [80] which is referenced in Figure 2.6.

Multiple changes have been performed, most specifically, to the sub-categorizations of each question. **Why** has been branched into **Actions**, and **Targets**. The former defines user goals and is structured in three levels **Analyze**, which holds **Consume & Produce**, **Search**, and **Query**. Within this category, everything is similar to the previous typology except for **Produce** which now has **Annotate**, **Record**, and **Derive** properties. Whereas, **Targets** express features of interest to the user for all types of data and attributes.

How keeps the **Encode**, although expanded, and **Manipulate** categories and additionally has **Facet**, responsible for modifying the visualization space, and **Reduce**, simplifies the complexity of the data. Compared with the previous model, **Manipulate** lost some of its methods and they got separated and added to **Encode** and **Reduce**.

The last question, **What**, displays a list of possible data types and types of attributes that can be rendered in a visualization. In the **Dataset** section, there are various types of **Data**, and **Data Structures** that can be rendered into a visualization. The other half is comprised of **Attributes** and their types, categorical or ordered (ordinal or quantitative).

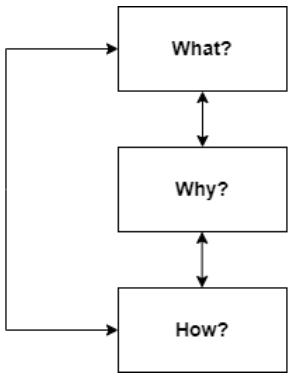

Figure 2.6: **What**, **How**, and **How** framework, taken from T. Münzner [80].

2.5 Visual Analytics

Combining the principles of visualization and data analysis achieves visual analytics. Visual analytics is the concept where the interaction between Humans (users) and the visualizations gains relevancy, due to the fact that the users are responsible for analyzing the data. Human analysis is beneficial since human knowledge, sense of perception, and intuition can not be replicated by automated systems. Thus, creating interactive interfaces with which users are capable of manipulating data and guiding the visualizations, will yield results that automatic analysis would not be able to accomplish. It encompasses various branches of the visualization field (from Information Analytics to Scientific Analytics), human cognitive studies (from Interaction to Perceptual Sciences), and data analysis (from Data Management to Knowledge Discovery) [54].

By definition, visual analytics is a field that leverages visualization, data analysis, reasoning, and human interaction to develop knowledge and recognize patterns in complex datasets, that computers might find problematic to make decisions in. Where visualizations and interactions are created to enhance the human cognitive and perceptual abilities [26].

Additionally, visual analytics depends on its iterative cycle that oversees the visual analytic process. Characterization of this process is the dichotomy between the computer and human processes. Although they create a feedback loop that involves both parties, it is nevertheless important that a distinction is made so it becomes explicitly obvious what each unit does. The computer is responsible for handling data, creating/displaying visualizations, and executing the analytical models. While humans intervene in these processes by transforming data, adapting models, and manipulating visualizations based on hypotheses.

This makes sense, computers can hardly perceive implicit connections between data and the domain, and humans struggle to analyze significant amounts of data [98]. The cooperation of both parties creates the necessary link to achieve new knowledge through formulating hypotheses, by proving or refuting them, as seen in the knowledge generation loop from Figure 2.7.

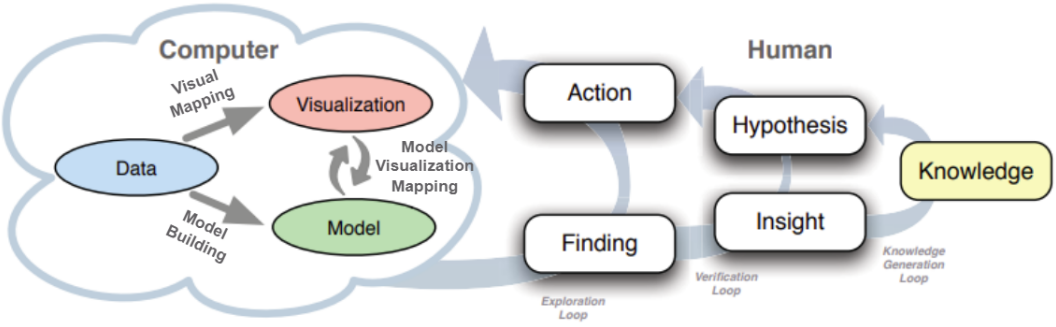

Figure 2.7: Knowledge generation model for Visual Analytics, adapted from D. Sacha et al. [98].

This results in better analysis and helps users with data interaction and data exploration. Additional knowledge and better analysis can significantly raise the veracity of results, shape the necessary basis to draw upon conclusions, and boost decision-making soundness [55].

2.6 Visualization Tools

The growing accessibility of data visualization has been extremely positive to the domain. Encouragement to learn and understand visualization literacy from a non-expert perspective [7, 4] can not be understated. Even though creating individual charts might have become an easy task nowadays it is still far from the complexity of creating intricate, however, effective, stories from data. New tools and systems are able to express most users' needs in simple and automated ways. Nevertheless, visualization tools range from manual, requires users to be versed in programming and visualization, to automatic, where almost no user interaction is needed. Somewhere in the middle, we find semi-automatic tools, where user interaction matters and visualization proficiency is advised [138]. One can assume that having high knowledge regarding visualization is almost a requirement in order to extract the most out of these tools.

On the other side of the spectrum, there are libraries, most of them supported in Python and R which is expected, seeing their data processing prowess. Libraries have a different approach when it comes to functioning. Besides providing default templates and built-in methods/functions that allow users to build and personalize visualizations, there is a heavy hand-made component that requires users to have previous knowledge about programming, data visualization, and, in some cases, data analysis.

Open-endedness awards a lot of flexibility and creativity to the user but the main drawback is the existence of a steep learning curve.

2.6.1 Tools

These systems attempt to address the inherent gap between common users and visualization experts.

Lack of literacy has been a known issue to the visualization community. A 2015 survey [17] interviewed 273 participants and judged these same participants in their ability to read visualizations, name them, and offer an adequate example in which the current visualization made sense. The results concluded that common individuals are familiar with a rather short list of graphical visualizations (Bar chart and Line chart) yet this knowledge rarely translates into favoring analysis of unfamiliar/unknown displays.

Another study [18], conducted in 2014, performed tests, on online platforms, regarding visualization literature in line graphs, bar charts, and scatterplots. The main objective was to identify the "ability to use well-established data visualizations to handle information in an effective, efficient, and confident manner", as described by the authors. Results

showed that individuals who attended the test were considered below average, but close to average, when looking at line graphs (can be a reflection of the simplicity and familiarity with line graphs), and below average when evaluating bar charts and scatterplots.

Understanding how other visualization tools combat these issues provides insight and a tangible guideline for the development of this dissertation's tool. Tools like Vega, Tableau, and Datashot are examples of visualization tools that represent different doctrines within the visualization domain.

- **Vega** (Vega-Lite) - is a grammar-focused tool, that bases itself on Wilkinson's "Grammar of Graphics" [129]. Created with the intent to generate web-based visualizations. These visualizations are defined by a JSON-like format, which means that users must know programming basics, and it offers users a multitude of possible designs. It can integrate Voyager [131], a data exploration tool that combines manual and automated approaches. It is an open-source project which features the browsing of visualization recommendations, suggestions of related views, wildcard specifications (specify multiple charts in parallel), a customization interface for visualizations, and a bookmark gallery where a user can store his favorite visualizations.

- **Tableau** - Example of an industry-standard tool, it is GUI-based and mainly works by dragging and dropping items. Its in-built data analysis and algorithms features help massively the interpretation of visualizations while offering a customizable foundation. Tableau's most distinctive component is "Show Me" [71], a visualization recommendation system based on available visual mappings of selected data from the user. Tableau also incorporates another important feature, "Ask Data", that enables the user to create visualizations using common language mixed with domain-specific terms, found in the input data schema. The user builds queries, in a common language, that are interpreted, given context, and translated by the system and receives appropriate visualizations for his queries, as a result. This feature provides the means to automatically build views in Tableau, even for users without past experience with data visualization tools.

- **DataShot** [123] - is an automated tool that creates fact sheets from raw data. The system has both data manipulation, data analysis, and visualization design techniques integrated. It is capable of making its own data exploration and automatically mapping data accordingly. All of this is without any kind of human interaction.

2.6.2 Recommendation System as Tools

The growth of data science and data analysis had a direct correlation with the increasing amounts of data that are kept for further analysis. Tools with the ability to explore data, find patterns and trends and underlying connections between data elements are necessary considering how time-consuming performing these tasks, manually, is. The objective of

visualization recommendation systems is to guide users by suggesting visualizations akin to their current analytical task [52].

The effectiveness of visualization for a certain analytical task depends on factors like data characteristics, domain knowledge, accuracy, user requirements, and user preferences. Creating systems that are fully capable of taking these factors into consideration is the final objective when thinking about visualization recommendations. Other two important characteristics of recommendations are the **Relevance**, in relation to the current context, and **Surprise** effect if the visualization has additional unexpected features [118]. Finally, it is needed to take into consideration how we present the suggestions to the user.

Literature shows us the popularity of these systems, even though they differ considerably from one another. Typically recommendation systems are inserted into one of three categories [138]:

In **Data-Driven Models**, data features are used to make predictions correlating them to similar data objects. These models have an interesting state-of-the-art, especially connected with machine learning. This recommendation system field separates itself from others that resort to guidelines and rules to generate visualizations. Machine-learning-based systems learn the relations between data and visualizations by models. VizML [49] is a machine learning system that learns based on choices that analysts would take (human-annotated ground truth). This is significant because it tries to predict experts' decisions. The authors even evaluate that trained neural networks can achieve similar test results on analytical tasks when compared to individuals. The possible availability of a model like VizML being open-source could mean the utilization of a data-driven approach, in our tool, that otherwise would be impossible due to lack of data.

Knowledge-Based Models use explicit knowledge models as a basis for prediction, by applying knowledge-based techniques to visualization design. This category uses metrics, guidelines, and rules in order to achieve clarity in which choices will be taken when recommending the visualizations. Although efficient and tailored for user preferences, the necessity of having rules can be a limiting factor. VizRec [118] is one of these models, that addresses this issue by having new datasets being analyzed by new users and by evaluating models that we developed to capture visualization utility. It works through online and offline processing of data, approximate query processing, and efficient ranking techniques to implement "on-the-fly" recommendations.

Hybrid Models are made of a combination of both Data-Driven and Knowledge-Based models in the interest of achieving advantageous analytical results. DeepEye [69] utilizes an online/offline methodology responsible for training machine learning models, on the offline end, where the first represents the decision tree responsible for mapping data to visualizations and the second ranks visualizations. On the online end visualizations are constructed in accordance with the decision tree, and then used as input for the ranking model. The outputs are the most relevant visualizations according to the machine learning model.

2.6.3 Recommendation System Testing

Aggregating the effort to research, test, and evaluate these systems individually creates a deep knowledge regarding these technologies and how they approach the generation of visualizations or recommendations for design choices. It is important to recognize that each system operates differently, with their respective strengths and weaknesses. In order to make a versed decision, this research is one of the most vital sources of information, capable of singling out individual systems. By focusing on a total of eight recommendation systems, that were involved in this process, these are the resulting insights:

2.6.3.1 VizML

The first system picked and consequently tested was **VizML** [49], it is a machine-learning-based model, trained with a one million visualization corpus, capable of classifying and creating visualizations. The model functions based on fine-tuned training through what the authors calls "Data Features". These features describe, individual or pairs, of columns from a dataset and allow the model to make decisions about the visualization in general (both the mapping and encoding of data into visual variables), pertaining to knowledge acquired from the data such as dimensionality, column type, statistical features, and so on. After training and inference setting, the model is then ready to make prediction tasks through its classifiers.

The identified system pros and cons were found to be:

- **Pros:**
  - **Training-based Visualization :** The system utilizes features as both a learning unit and as visualization design characteristics. Training comes as an integral part of the model that projects the acquired knowledge as output visualizations.
  - **Ability to rank and classify :** Besides generating visualizations, VizML can theoretically classify them and simultaneously rank them amongst the top-K, K being a determined number, created visualizations.
- **Cons:**
  - **Missing Resources :** Since the project was not updated recently, and the website of the project was shut down it was impossible to retrieve its dataset and model inference. Hence, the model could not be tested since all machine-learning-related resources were unavailable.

2.6.3.2 Data2Vis

Data2Vis, one of the alternatives, that was presented as an extension of the VizML use cases [33], is a recommendation system based on a **Transformer** model named **Seq2Seq** that works based on tokens (words) that slowly build relationships between each other in

an effort to build context [112]. The system assimilates the model through the handling of a declarative grammar that is accepted as input and is also generated as output. More specifically, the data is parsed and translated to an intermediate language that is both consumed and generated by the model, the latter being a specification of a Vega-Lite visualization [104], seen before in section 2.6.2.

Unlike the previous model, Data2Vis was able to be tested and it presented these significant points:

- **Pros:**
  - **Low Data Training Needed :** Data sampling was used as a way to enable effective model training even with a relatively low, in comparison with other data-driven models, training dataset.
  - **Lightweight & Versatile :** Rendering of visualizations was made to reduce the cognitive load while still being expressive, in comparison with tabular data when represented visually.
- **Cons:**
  - **Limited Visual Options :** The available quantity of visualization diversity was lackluster and only a fraction of the visualizations our team hoped to achieve. A dataset that was small-scaled and diversely scarce contributed to these results.
  - **Non Extensible :** Non-existing support for adding features to the model, such as new defining preferences or user requirements which is one of our primary objectives.

2.6.3.3 KG-4 Vis

Data-driven tools, including **KG-4 Vis** [64], are typically machine-learning focused but sometimes mixed knowledge-based principles, commonly called in the literature Hybrid Models. This system incorporates a knowledge-graph approach to learning, since they are structured to represent concepts of the human knowledge, yielding them able to provide recommendations. Akin to VizML, data/column features are used for classification. Works through turning data features in knowledge graphs where each feature represents a vertice, and then creating mapping relationships between features and design choices for individual visualizations. Said knowledge graphs are transformed into embedding vectors, TransE [15], where a head (H) and tail (T) are connected by a relation (R) in a triplet (H,R,T). To conclude, the inference of embeddings is calculated to determine how to map a feature, or numerous features, into a visual variable. As a result, the system is highly extendable and customizable by mixing data with pre-determined rules.

Found spurs and hurdles for the respective system:

- **Pros:**

  - **Adaptable :** The system is highly adaptable due to the inclusion of rule-based generation that influences visualization recommendations in cooperation with the model-resulting inference.

- **Cons:**

  - **Limited Visual Options :** The authors explain that it was a design choice to stray further from visualizations and encodings that could be highly specific and required professional knowledge [64].
  - **Inadequate Training Results :** Obtained results were far from ideal since the recommendations were all very similar with a lack of diversity, every visualization looked identical to the ones previously shown.

2.6.3.4 Draco

Afterward, **Draco** [132] was tested with good/promising results. Draco's model falls under a different category from other systems, called: **Knowledge-Based Models**. Metrics, rules, and constraints are defined as fully customized sets of assertions that guide the model's preferences. These rules are written through an **Answer Set Programming (ASP)** language called Clingo [38] and they are responsible for characterizing everything within the model itself: the fields, the tasks, and the visualizations.

The system functions by receiving information about available data, information regarding data, and visual elements (e.g. marks, scales, and encodings) following the same aforementioned assertion format. Each assertion has a pre-determined weight, this weight is added to a sum if the assertion is violated, and the system returns the lightest, therefore the chart with less calculated weight, as the answers to a specific instantiation. The system then produces visualizations by translating single instantiations into Vega-Altair [117], which is a declarative statistical visualization library that utilizes the Vega-Lite [104] engine, correctly defined visualizations. Besides, the ASP language gives the system a highly adaptable component that works seamlessly with some of our fundamental premises. Also forming a unique, although simple, independent format for describing visualizations is a key component that might help us build an automatic pipeline. Nevertheless, the model showed both advantages and disadvantages, for instance:

- **Pros:**

  - **Specific Visualization Format :** The system utilizes a "fact-centric" format to interact with the internal model. Formats like these facilitate automation and express knowledge-creation using knowledge repositories, and are capable of positively integrating Large Language Model (LLM) technologies, where the LLM builds partial or full specifications to be later analyzed and rendered as visualizations.

  - **Ranks and Recommends Visualizations :** Capacity to evaluate, rank, and recommend visualizations is the main feature of the system, this evaluation and ranking is done through the calculation of the weight regarding visualization after being processed by the internal model.
  - **Adaptable :** Constraints and the answer set programming model elaborate an easy implementation foundation to create abstractions and determine preferences over the design choices and data processing methods.

- **Cons:**

  - **Requires Training :** Since Draco allows for the inclusion of rules through ASP, it is possible to abstract over to the system the concept of analytical tasks and assign a specific behavior for each. To be able to do so it is crucial to form a basis, set the possible tasks, and then give them specific rules. Thereafter, there will be a requirement to train such data for the model to be able to assign these new rules their respective weights. This would require the creation of a handcrafted dataset, although due to the knowledge-based nature of the system, the dataset could be significantly smaller.
  - **Limited Visual Options :** The system's existing chart types are sufficient, but there are few options when it comes to encodings and the limit of fields used, per visualization. Mainly, the number of available fields to obtain a good visualization is more constrained because of the cardinality of usable encodings.

2.6.3.5 DeepEye

Leveraging the two common types of recommendation systems data-driven and knowledge-based models creates a new type of mode: Hybrid Model. **DeepEye** [69], one of such hybrid models, recognizes, classifies, and more importantly recommends visualizations, taking into account graphical diversity. The overlay of both types of systems creates the necessity of having the system split in two: **Online** and **Offline** elements.

Amongst these components, there is the machine-learning training part for which the offline department is responsible, while the online department tries to identify the most adequate visualization possible, taking into consideration the data presented to the model. It tries to encapsulate how humans' perception of visual variables brings about finding intuitive visualizations. Lastly, this system reckons the possibility of choosing from three available ranking models: Learning-to-Rank - Used with the machine-learning (Offline) model to rank visualizations. It is a supervised learning task that takes feature vectors as input; Partial Order - Collection of standard rules that define a ranking principle and build a graph where each vertex is a visualization; Diversified Ranking - Selects diverse visualizations to reduce redundancy, similarly to Partial Order it utilizes a graph-based methodology.

The main concepts concerning the model are:

- **Pros:**
  - **Multiple Visualization Options :** The system offers a commendable amount of visualizations choices and diversity. Resulting outputs come in a variety of visualizations that are highly comparable to human-made visualizations.
- **Cons:**
  - **Non-Extensible :** The model does not seize the possibility of incorporating user/expert agency, neither via the inclusion of the user's preferences and/or the expert's abstractions, since only the API is publicly available. Thus, it is impossible to take into consideration our formal proposal.

2.6.3.6 Voyager

Voyager/Voyager2 [131, 130], not only a recommendation system but also a data exploration framework, designs and ranks visualizations based on statistical values that originate from aggregating all entries and then evaluating the data exposed to the model and how it would be distributed on each chart. As a data exploration frame, it suggests fields, interesting operations (e.g. aggregations), and allows for dataset exploration as a means to acquire higher knowledge. This system's interactions can be executed **manually or automatically**. Another specific feature of the system deals with related views/partial views, besides the data exploration influence it could be of interest when analyzed through the data visualization specter.

Testing in the visualization tool was done through the available embedded web application library, but it would not be useful to this project, since I was unable to effectively run the source code locally. Regardless, the results obtained were positive although heavily focused on the data exploration aspect.

- **Pros:**
  - **Encouragement of Data Exploration :** Since Voyager is described as a data exploration framework, this mark is palpable within the system itself. Emphasizing partial views, interesting data fields, and "group by" operations offers a new level of abstraction to the user.
  - **Manual and Automatic Approach :** The system approaches data visualization generation with two available methods. An automatic method where the visualizations are created following the model inference, and then the manual one where the user is able to drag-and-drop or exchange visual and data elements.
- **Cons:**

- **Unable to Execute Source Code :** Was not possible to work with the source code of the system and run it locally, which would be necessary in order to customize the code base, thus making it non-extensible.

2.6.3.7 ZenVisage

Similarly to Voyager, **ZenVisage** [1] is a data exploration framework that serves as a recommendation system simultaneously. Along with visualizations, it is also capable of suggesting data queries to obtain said visualizations. It assimilates knowledge through a sketching and pattern-matching strategy. The input is given to the system via a specific query language, named Zen Query Language.

- **Pros:**
  - **Recommendation of Queries and Visualizations :** Theoretically, the system provided both recommendations of possible data queries, after thoroughly analyzing the dataset, as well as the designated data visualizations.
- **Cons:**
  - **Unable to Execute :** Due to a lack of support for the tool, it has become deprecated and impossible to execute tests on. Some of the designated libraries are no longer available and this rendered testing impossible.

2.6.3.8 Table2Charts

Table2Charts is a framework with data queries and design choices created from data tables, that used VizML, DeepEye, Draco, and Data2Vis as baselines for comparison. It is a **Data-driven tool that turns visualization recommendations into table-to-sequences** with next-action-token estimation to fill visual design choices. This reasoning conceives the ability to make partial recommendations through the utilization of already existing data queries and design choices. These partial recommendations pass into the heuristic calculation component, with one shared encoder and multiple decoders, and provide recommendations at the end.

The system presented the following points:

- **Pros:**
  - **User Requirements as Input:** The system worked with an input that took into consideration data queries, data to be analyzed, and visualization design choices, how the data should be drawn [136].
  - **Recommendation of Queries and Visualizations :** The functionality of generating data queries and visualizations is available within the system. In the case of single-type tasks, the system suggests auto-fill and completion of details for

a chosen chart, whereas in multi-type tasks the model helps the viewers by establishing multiple possible visualizations to aid the user analyze the data table.

- **Cons:**
  - **Missing Resources :** Although the system looked promising and had a very interesting proposition of comparing the results against other models, that were also researched and tested in this dissertation. The system combines the VizML dataset, which was already established as non-attainable, and a dataset composed of HTML data tables. Unfortunately, it was not possible to test with only half of the data.

After testing the aforementioned models, at least the models capable of being correctly tested, the conducted research was concluded by creating a table that reflected knowledgeable information regarding each system. Figure 2.2 was inspired by the tables presented by VizML and Table2Charts teams [49, 136] and condenses an overview of information.

System	Learning Tasks	Type Model	Training Data	Data Generation (Made by)	Vis Results
VizML	Classification Tasks	Data-Driven	Visualization Pairs	Users & Experts	Not Applicable
Data2Vis	End-2-End Visualization Generation	Data-Driven	Visualization Pairs	Rules & Validation	Positive
KG-4 Vis	Knowledge-Graphs	Knowledge-Based	Entity Relations	Users & Experts	Negative
Draco	Hard & Soft Constraints	Knowledge-Based	Good/Bad Chart Pairs	Rules & Annotations	Positive
DeepEye	Data Classifier & Ranking	Hybrid-Based	Good/Bad Chart Pairs	Rules & Annotations	Very Positive
Voyager	Statistical Data Classifier	Data-Driven	Not Specified	Users & Experts	Positive
ZenVisage	Graph-Based Query Language	Data-Driven	Relation Data	Users & Experts	Not Applicable
Table2Charts	End-2-End Visualization Generation	Data-Driven	Vis and Tables Pairs	Users & Experts	Not Applicable

Table 2.2: Overview of the researched recommendation systems, inspired by Hu et al. and Zhou et al. [49, 136].

2.7 Modeling

Models usually refer to artifacts expressed through a modeling language. Models draw abstractions that represent a real system. This creates an opportunity to translate a complex system into a digestible diagram that grants the possibility to make predictions and reason about said system [60], by means of declaring their properties and rules in a uniform and strict manner, expressed by precise languages. These languages are defined by meta-models, which are models that instantiate other models, capable of constructing detailed syntax and semantics. We can think of models as interpretations of entities

through some specific languages while meta-models function as the set of instructions that validate the statements of those same languages.

2.7.1 Model-Driven Initiatives

Model-driven initiatives have evolved a lot since the usage of models has become increasingly bigger. Modeling in engineering helps in the development of artifacts, typically other models, that hold information and can be used in the production of everything from system specifications to fully-fledged software [84]. In Figure 2.8, we can see the current landscape of the model-driven software methodology.

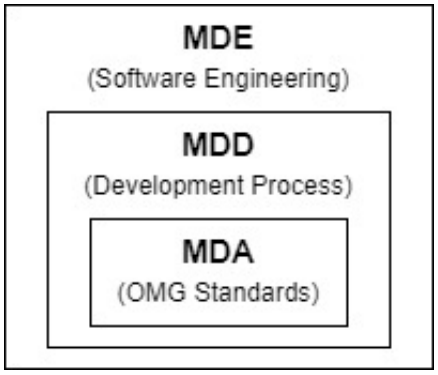

Figure 2.8: Model-driven hierarchy, adapted from D. Ameller MA thesis [9].

2.7.2 Model-Driven Engineering

Model-Driven Environment (MDE), shows itself as a software development methodology capable of addressing domain concepts by combining "Domain-specific modeling languages" and "Transformation engines and generators" as a means to abstract and synthesize, respectively, these concepts into domain models [105].

As a subset of MDE, Model-Driven Development (MDD) is a development paradigm that sees models as a primary foundation block during the system development process. Providing high-level abstraction, reducing the system's complexity, and remaining mostly platform-independent are crucial aspects that characterize this paradigm. These abstractions allow for the creation of model specifications that grow away from the domain problem but remain highly adaptive to new environments. Enabling developers, during the development stage, to build models that specify the system conditions, functionalities, and architecture while providing a clear path towards the end goal by maintaining the models as state-of-the-art as possible [11].

2.7.3 Model-Driven Architecture

Model-Driven Architecture (MDA) [84], is Object Management Group (OMG)'s particular version of MDD that relies on the OMG standards. The standards support models to be represented in multiple modeling languages, to be transformed, and to execute in different

abstraction levels and domains. One of the primary objectives of MDA is to express data into models that can be used further on as objects of study (queries, analysis, reports, transformations, etc.). By its standardization, MDA models can echo a lot of knowledge about the implementation of a system that can be improved by successive iterations.

According to OMG, MDA is divided into ten basic concepts: System, Model, Modeling Language, Architecture, View and Viewpoint, Abstraction, Architectural Layers, Transformation, Separation of Concerns, and Platform.

System - set of concepts and connections that can be organized to fulfill a goal; **Model** - relates to the system by mapping its characteristics into an artifact; **Modeling Language** - presents the language upon which the model is built, these languages have a wide range of possibilities; **Architecture** - tries to better define systems; **Views and Viewpoints** - create notions about the construction and selection of the system information that becomes reusable to multiple systems; **Abstraction** - allows for an easier understanding of the system and its details/integral concepts; **Architectural Layers** - identify the different abstraction layers of an architecture; Transformations - produce models by claiming one or more models as input and getting one or more target models as output, this happens by applying transformation models.

When applying transformation models is paramount to have a deep understanding of syntax and semantics in both models [original and produced] since they are crucial in the process of transforming Platform Independent Model (PIM) into Platform-specific Model (PSM) [107]; **Separation of Concerns** - an important feature of MDA that makes it possible to separate different aspects of a system by themselves, without depending on other components. Different specific viewpoints and transformations establish Separation of Concerns as a possibility.

The MDA architecture has three types of models: Computation Independent Model (CIM) - the first model built, using object data; Platform Independent Model (PIM) - a model expressed by a modeling language without an explicit relation with any implementation; Platform-specific Model (PSM) - a model expressed by mapping a PIM into a specific implementation language or platform [90].

In terms of modeling environment, MDA is broken into four levels. The lowest, M0, where only object data exists. After that comes M1 where the PIMs and PSMs reside, these models instantiate attributes, methods, and relationships. The third level, M2, is characterized by the Unified Modeling Language (UML) standard. Finally, the last level M3 is comprised of the Meta Object Facility (MOF) which is a meta-model of meta-models [25].

2.7.4 Modeling Languages & UML

Languages are a central part of modeling, but the possibility of having specific syntax and rules created diverse languages tailored to different spotlights. Adopting a language before any advancements in the process of building a model is a crucial step since it will dictate

the model's approach. The two main types of modeling languages are General-purpose Language (GPL) and Domain-specific Language (DSL).

GPLs gravitate toward being more generic, resulting in poor support for accurately translating certain specifications and/or notations within a certain domain. DSLs are generally more expressive, specific to a reduced number of domains, but offer better abstraction to their respective domains [59]. In conclusion, the intrinsic distinction between GPL and DSL is their specifications. DSLs are effective in often limited sets of tasks while GPLs are supposed to be more generic and complement tasks across multiple application domains [51].

UML [86] is a general-purpose [visual] modeling language, essentially it is a meta-model to build other models using a strong visual component through the usage of diagrams. An important UML feature is the availability of broadening the UML basic functionalities by admitting the creation of more generic or more domain-specific capabilities. UML was standardized by the Object Management Group with the intent to design and implement software-based systems in an improved way. The UML standard, level M2 of the MDA modeling environment, enabled the design of a wide spectrum of models to be drawn in a way that promotes simplicity, readability, and the exchangeability of information between different models.

2.7.5 Conceptual Modeling & Conceptual Models

Modeling is a technique that focuses on bringing knowledge and consensus, based primarily on abstraction and modularity [73]. Conceptual modeling, sometimes referred to as "Modeling with concepts", allows the building and reasoning of conceptual models. These models encompass active modeling in fields like database design, general software, business processes, and many others [42]. It defines concepts relative to the real world in the pursuit of offering better knowledge, communication, and problem-solving about these specific domains [81].

Conceptual Model (CM), follows a conceptual methodology. Abstract over aspects of specific domains and serve as a point of relation between relevant concepts and the domain itself. The models are based on a formal notation that is intended to be, above all, made for human comprehension [81].

Conceptual Modeling is a sound strategy to design and develop efficient systems, especially in situations where the complexity of data is substantial. The conceptual models facilitate the generation of systems that support decision-making processes within the expected domain [94]. Although, certain domains are under constant evolution and therefore CMs must also be capable of adapting easily and provide improvability.

CMs are particularly good at supporting Information System (IS) since they need knowledge about a domain before being able to perform specific functions that might be required by the system. Most ISs have three main functions: Memory function - maintains the representation of state of the domain; Informative function - offers information about

the state of the domain; Active function - executes functions that change the state of the domain. [85].

The development process of ISs [81] requires knowledge about:

- **Subject World -** consists of the subject matter for the system. It represents the information system and relevant data within it at the different levels of abstraction of implementation, ranging from functional requirements, to model designs or implementation.
- **System World -** contains the implementation layers, from conceptual to functional.
- **Usage World -** the domain of the system that will try to abstract and function upon, using multiple components like agents, activities, tasks, projects, and users.
- **Development World -** describes the process that led to the development of the information systems, the team of analysts, methodology, their design decisions along with the respective justifications behind each.

These four "worlds" are interconnected by virtue of depending on each other. Users (usage world) need developers (development world) to build systems (system world) that keep information about the domain (subject world). All of this knowledge has to be shown in a way that can guide and aid the process of building information systems and this is where CMs come into the equation.

2.8 Conceptual Models

The modeling of a domain is not a simple activity. It requires expertise and solid reasoning to achieve a satisfying result that accurately abstracts upon the original domain, in a complete manner. Thus, resulting in a model that communicates a new layer of insight and allows for deeper analysis surrounding the domain [73]. One of the main features of our tool is the capability of considering the domain model (a conceptual model) as input data in the process to achieve the necessary abstraction to map it to existing analytical tasks, although middle steps in between the mapping process will not be executed by the CM. This means, hypothetically, that this tool would be able to analyze any specialization independently of the domain, assuming that the model fulfills all crucial requirements of domain abstraction.

Regarding conceptual models, it is clear that there has been a considerable amount of interest from research communities to incorporate conceptual modeling into their studies and projects. Areas that benefit from CMs include Requirement Engineering, Ontology Studies, Knowledge Representation, and Object-Oriented Models [27]. Extracting system specifications, defining information for information systems, abstracting terms or concepts to discuss domain-specific problems, and improving domain knowledge are just some of the approaches that can be found in the literature, i.e. [95, 68, 43].

More specialized fields, that rely on multiple core concepts (i.e. the example seen in Figure 2.9), are also keen on experimenting with CMs to possibly aggregate particular parts of the domain into a whole.

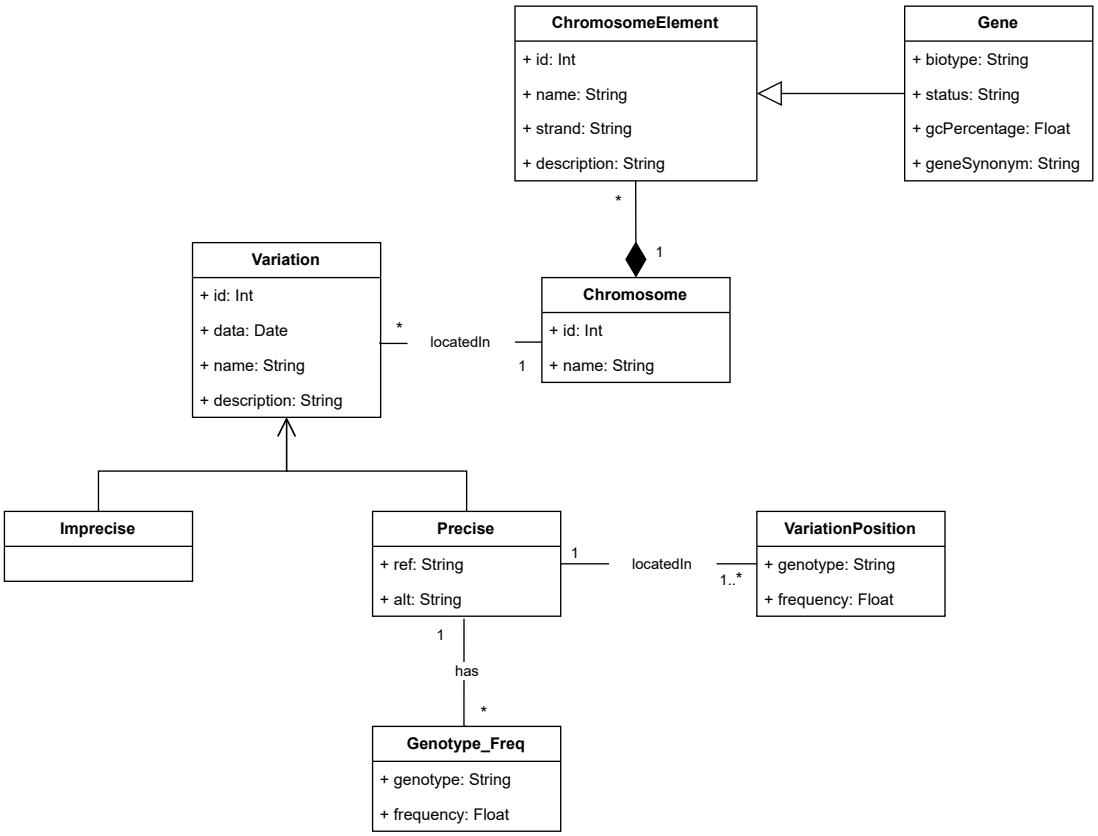

Figure 2.9: Conceptual model of a human genome domain, adapted from Léon A. (co-advisor).

This is justified because the structure of these models allows for iterative evolution of the model, high modularity, and overall simplification. An example can be seen in [63], this specific case occurs in the domain of genomics, where the model's development is made by interconnected parts of different concepts, originating from genome and protein repositories/databases, that can be connected to acquire the complete model. The usage of CMs in such domains is valuable since the model can be built upon structured data, that already exists, originating from distinct databases that might have remarkably specific concepts. Another study promotes the idea of using conceptual models to proliferate information-seeking processes to researchers within Computer Science fields [72], the novelty relies in the deliberate existence of visual models since previous models were purely textual. The use of conceptual models allows for researchers to understand carefully and without ambiguity how the information is searched and kept. The aforementioned studies show the increasing popularity of CMs, their relevancy, and ultimately their usefulness.

Just like many other models, conceptual models also need to be characterized. The

formalization of a framework capable of characterizing conceptual models [30] proposes a structure focused on the activity of developing conceptual models, and it was composed by the identification of the model goal together with three dimensions axes: **Timing** - When was the model done during the development stage; **Automation** - How much automatic assistance does the model provide; **Genericity** - How generic/abstract is the model.

Although the framework goal was achieved, the authors decided to revise and rebuild the framework focusing on fixing limitations [29]. The first framework had issues with classifying certain types of models, research models to be more exact. Therefore, the latter had to resolve such issues. The new framework is divided into three sections. Firstly, the **Context** table contains the purpose, intended domain, domain of study, and intended users. Secondly, the **Models & Languages** matrix consists of conceptual models (new or existing), patterns and meta-models, and the modeling languages. And lastly, the **Tools & Processes** matrix which included methods, and automated tools.

Fundamentally, the existence of literature that features conceptual models in unison with data visualization is very scarce. This is a significant consideration and a motivating factor as to why this dissertation is differentiating itself from other alternative works in research.

Conceptual models are great at simplifying and communicating information, thus using CMs as a method of describing visualization systems seems logical. Following this reasoning, Moral et al. [76] proposed a model that contains cornerstone concepts of visualization, devices, visual representation, and analytical tasks. Due to the high abstraction levels, scalability, and flexibility, this model adapts without much effort to any domain. Trying to build such a holistic model of this complexity is an arduous task. The approach, although similar to ours in the sense that it involves conceptual modeling, it tends to follow an idea of being more of a framework capable of describing visualization systems. Even though this is not our objective it still shares a lot of common ground regarding modulation, and how the mapping of concepts between data, analytical tasks, and visual elements works.

One of the expectations of this project is to provide a new look into how models, more specifically conceptual models, can create a solid foundation for which data (raw real-world data) can be correctly processed into valid domain concepts and build sturdy and data-analysis-ready datasets, their structure, their data types, and their contents.

3

Taxonomy

3.1 Introduction

One important component of this dissertation is the possibility of leveraging **Domain Questions** in furtherance of obtaining the necessary **Analytical Tasks** to conduct a capable answer, to the original question.

Looking through the prism of existent taxonomies, **Analytical Tasks** act as the main pillar of any data visualization taxonomy. These can be viewed as data-oriented actions, divided into high and low-level tasks, that enable interactivity from the user in an attempt to attain meaningful knowledge [102]. For this reason, our team decided to build its own standard definitions:

- **Domain Question -** Represents a question about a specific domain that the end-user wants to answer.

- **Analytical Task -** Abstraction that determines an action, applied to data, by which the user might achieve a total or partial answer.

Even though the analytical tasks are quick access procedures to aid the user in interacting with the system, our taxonomy mainly targets processing tasks that the user can execute solemnly on data. Meaning, that this taxonomy is not interested, at the time of writing, in covering tasks that interact directly with the display. For example, we [Team] initially thought about including the paper "The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations" [108]. However, the strong interactive component, (i.e. Overview, Zoom, Details-on-demand, and History), dictated that it would not be included in our research.

The terms **Item** and **Element** will be used interchangeably during this chapter to relate to entries in the dataset. Furthermore, the term **Attribute** will be connected to the individual attributes within each item/element, and the term **Characteristics** will link to non-attribute characteristics of each item/element (e.g. position, in the visualization space).

3.2 Investigation & Research

Research began by analyzing extensively the existing literature related to typologies and taxonomies, within the scope of data visualization. Firstly, the laying of a solid foundation is imperative, thus selecting three diversely quoted taxonomies - in the data visualization literature - were our starting point: Münzner's work [80], a three-level typology, Wehrend & Lewis [127, 116], and Amar et. al [8], the latter works being low-level taxonomies.

The full collection of surveyed works was assimilated with various publications of renowned literature within the scope of data visualization. However, some of them, e.g. Quadri and Rosen's survey [92], were found through associative searching because they referenced or addressed one of the three core pieces of our survey. Therefore, we utilized these works to lay a sound foundation and serve as a reference-point to other works that shared a focus similar to the one we were trying to achieve. This taxonomy proposal was supposed to create a specific environment to perfectly fit our approach by providing it with analytical tasks that were deemed necessary.

Before analyzing these taxonomies further, it is important to clarify that some taxonomies will not be featured in their entirety since they involve interactive tasks. Additionally, tasks that rely on geospatial data like "Geospatial" and "Spatial Relationships", mentioned in Yau's [133] and Börner's work [16], respectively, will not be included because spatial, and temporal data, as well as different types of structures (e.g. nodes), are already accounted through our meta-model. As such, it is not necessary to specify analytical tasks for individual data types.

Wehrend's work [127] discusses a detailed characterization of possible operations, under the label of **task**. Being a low-level taxonomy, with a diverse set of minimalist tasks presented has a schema to classify analytical problems [116], it is comprised of:

- **Locate**: The user possesses previous knowledge about the data and can locate and/or identify it.
- **Identify**: The user is capable of describing the element, even if he does not know about its presence.
- **Categorize**: Allows for categorizing and differentiating data through user-defined categories.
- **Cluster**: Groups elements that share relations (e.g. categories, attributes).
- **Distribution**: Presents a distribution of the elements determined by an attribute.
- **Rank**: Orders the elements within a certain classification, determined by the user.
- **Compare**: Individual, or groups of, elements can be compared between each other.
- **Associate**: Relationships between elements in a visualization can be established.

- **Correlate**: Identifies shared attributes in elements.

Münzner [80] presents an alternative in terms of concepts and rationale. The focus changed to a hierarchy of high, medium, and low-level tasks.

Münzner's framework has a small array of suitable tasks able to provide answers for users. This typology is characterized by the following tasks:

- **Analyze**: Users are able to analyze the data visually.
  - *Discover*: Discovers new knowledge related to the data.
  - *Present*: Displays visualizations graphically.
  - *Derive*: Generates new information from already existing data.
- **Search**: Searches and finds data based on knowledge, or lack of knowledge, with relation to either elements' location and/or properties.
  - *Lookup*: Defines the search it happens when a user knows the element being searched and where it is located.
  - *Locate*: A type of search conducted when the user knows what it is looking for but does not know the location.
  - *Browse*: The way of searching through a visualization where the user knows the general location of the element but not the element itself.
  - *Explore*: Search action characterized by the user's lack of knowledge about the element and its location.
- **Query**: After successfully conducting the **Search** task, grants the user to work with the found data. This level provides three lower-level functions:
  - *Identify*: Offers, back to the user, the attributes and characteristics of a specific element.
  - *Compare*: Bears comparisons between different elements.
  - *Summarize*: Provides an overview of the found elements.

Low-level taxonomies manage tasks in such a way that creates a strong language for higher-order externalizations of data within automatic generation systems [8]. These tend to mimic closely the system, where they are included, and be dependent on it. Amar's approach [8] supports the notion of establishing a task that has visualizations as the final result. By following this metric, tasks were determined based on their core knowledge goal (e.g. "What questions can a task address?"). Equivalently to our approach, the taxonomy is disengaged from any system-specific tasks (interactive tasks) and concentrates on ten cognitive tasks, as the author calls them. The aforementioned cognitive tasks are:

- **Retrieve Value**: Finds values of attributes that respect specific conditions.

- **Filter**: Given conditions on an attribute value help the user find elements that satisfy said conditions.
- **Compute Derived Value**: Calculates numerical/statistical operations (sum, average, max, min, etc.) on attributes.
- **Find Extremum**: Find the highest and lowest values of an attribute.
- **Sort**: Orders elements fitting to some ordinal heuristic.
- **Determine Range**: Identifies the range of values of a specific attribute.
- **Characterize Distribution**: Determines the distribution of values of a quantitative attribute of interest.
- **Find Anomalies**: Discovers anomalous values (outliers) concerning other existing attribute values.
- **Cluster**: Finds groups/clusters of related attribute values.
- **Correlate**: Resolves useful relationships between values of two attributes.

After evaluating these three initial works, we further analyzed other taxonomies in order to develop our research and accomplish a better and more complete proposal.

The resulting taxonomy of Valiati's paper [116] aggregates insight from other various fitting research [109, 108, 137, 78, 8]. The taxonomy brings together different levels of complexity in terms of analytical, cognitive, and operational tasks both in data visualization and exploratory analysis. Providing core high-level analytical tasks and intermediate tasks:

- **Analytical**: These tasks settle the creation of knowledge in an attempt to achieve a desired outcome.
  - **Identify**: Connected with finding, discovering, or estimating clusters, relationships, values, patterns, categories, characteristics, etc... The task finishes when a result is obtained or the question switches explicitly.
  - **Determine**: Describes calculating or defining statistical values. It is considered finished when a particular value is reached (e.g. variance, coefficient, deviations, hypothesis tests, etc...).
  - **Infer**: Highly linked to data analysis. After inferring new knowledge the user must create new questions or hypotheses.
  - **Compare**: Compares attributes and characteristics between items.
  - **Locate**: Defined by the actions of searching and finding recognized information, in the visualization.
- **Intermediate**: These tasks extend the analytical tasks and aid them in their final objective.

  - **Visualize**: Draws the data into the visualization space, by utilizing user input as base parameters.

Taxonomies in data visualization tend not to be categorized and this brings a new perspective on how a task might be organized and how the tasks cooperate with one another. Laha conceived such a categorized system divided by work operators (or goal-driven **tasks**), enabling operators (or supporting **tasks**), visualization parameters, and dataset properties [61]. Any task has a symbiotic relationship in a way that individual tasks affect the next task being performed.

- **Search**: Objectifies identifying a single particular element that can exist or not.
- **Pattern Recognition**: Searches for patterns such as trends and repetitions, either on a global or determined interval.
- **Spatial Understanding**: Generates knowledge about the position/display of data elements. These comparisons can be made between elements, in relation to the overall dataset or through the intersection of elements.
- **Quantitative Estimation**: Resolves quantitative estimates about specific element attributes (e.g. density, volume, length, size, etc.).

Only work operators were included, although one work operator task is missing since they are interactive-focused, unlike the remaining tasks. For our proposal, we are only taking into consideration data-oriented tasks.

Efficiency and accuracy are hot topics in data visualization, as Quadri and Rosen recognize in their survey [92], in the quest to obtain the best possible impact. An efficiency-modeled taxonomy based on the most common low-level tasks throughout the literature [92] is a design that aligns closely with our own proposal's idea. It would be significant to us, after all, high-level tasks can withhold diverse sets of low-level tasks hierarchically. This survey reports not only the tasks but also the visual encodings and the most appropriate visualizations. The last concepts will become relevant later in this chapter (section 3.8). The supported tasks, of this taxonomy, are the following:

- **Retrieve Value**: Given a specification it returns the attributes of elements that match the specification.
- **Compute Derived Value**: Creates statistical summaries (visual aggregations) from existing quantitative data.
- **Find Extremum**: Discovers edge cases within a determined range of values.
- **Sort**: Regulates the order of a dataset according to a heuristic.
- **Determine Range**: Determines the span of values within an interval.

- **Find Anomalies**: Finds outliers in data.
- **Characterize Distribution**: Characterizes the distribution of values for an attribute.
- **Cluster**: Identifies clusters of data that share similar attribute values or categories.
- **Correlate**: Recognizes useful relationships between element's attributes.
- **Compare**: Compares attributes between individual, or groups of, elements.

Andrew Abela [2] overviews a taxonomy with four types of tasks that can be executed by each type of graph. Combining these high-level tasks with the best available visualizations to accompany them is a crucial factor in both this work and our research. Although abstract their all-encompassing range creates a deep enough taxonomy for data analysis formalized by:

- **Relationship**: Shows how elements share relationships between themselves.
- **Distribution**: Identifies the distribution of determined values.
- **Comparison**: Compares attributes of two elements.
- **Composition**: Describes the dataset as individual parts or as a whole.

Yau's book [133] reflects on different types of analytical operations, the most interesting for our analysis being:

- **Proportions**: Resolves how individual parts relate to a whole.
- **Relationships**: Identifies relationships between element's attributes.
- **Differences**: Distinguishes characteristics between elements.

Few [35] refers to the importance of understanding the differences between the relations (typically described in other taxonomies as tasks) that can be displayed in visualizations. Adopting a categorization where tasks are distinguished as relationships between elements is an interesting switch on the standard concept of analytical tasks. The author suggested relations are:

- **Ranking**: Reveals how quantitative values can generate organized relationships, for example in ascending or descending order.
- **Part-to-Whole**: Makes segments that create a whole and compares the segmented parts to the whole.
- **Deviation**: Displays how a collection of quantitative values differs from a set of standard values.
- **Distribution**: Details how the quantitative values of an attribute are distributed.

- **Correlation**: Compares the variance between two collections of quantitative values. This correlation can imply a causal relationship between these two collections or the influence of exterior relationships.

- **Nominal Comparison**: Determines a comparison between discrete quantitative values using a non-ordered categorical scale.

Continuing our analysis, Börner [16] conducted a survey, similar to ours, in which he developed a visualization framework composed of six coordinates: insight need types, data scales types, visualization types, graphic symbol types, graphic variable types, and interaction types. Taking a look at these coordinates, we are majorly interested in "insight need types", also named basic task types. This is the most relevant component to our investigation. Hence, the authors identified seven tasks as a result of their study of existing taxonomies [35, 14, 133, 127, 93] and unified a nomenclature schema, namely:

- **Categorize/Cluster**: Categorizing refers to indicating items to categories of similar data, labeled by users.

- **Order/Rank/Sort**: Relates to ordering elements concerning each other according to a sequence or ranking heuristic.

- **Distributions**: Allows the user to observe how items are scattered in the visualization space.

- **Comparisons**: Process of analysis that establishes a comparison between elements.

- **Compositions (also of Text)**: Shows the individual parts of a disarranged whole.

- **Correlations/Relationships**: It refers to the process of expressing a relationship between two or more objects.

Finally, we took into consideration Li's work [66], which is a framework that focuses on the automation of visualizations and it is composed of four layers: requirements, characteristics, attributes, and visualization. Capable of mapping user requirements to visualizations, it implements a short taxonomy thought for a system that shares some key concepts with this dissertation. Our focus will be directed exclusively to the first layer, the requirement layer, where the resulting analytical tasks can be found:

- **Distribution**: Shows the user the location of a data item.

- **Composition**: Allows to determine the organization of elements in categories and hierarchies.

- **Comparison**: Authorizes comparison between attributes.

- **Relationship**: Discloses connections between different elements.

3.3 Analysis & Discussion

After conducting our research, it became apparent that the newly acquired information would be better analyzed through a unified structure that could integrate the diverse taxonomies and reveal possible common ground between their concepts. As such, we identified all the compatible tasks and possible task hierarchies between taxonomies.

3.3.1 Agreeing Tasks

During our analysis, some differing terminologies that converge into the common concepts were found. Hence in this section, tasks with similar execution and objectives will be grouped in an effort to achieve a comprehensive overview of all available tasks. Table 3.1 displays a comprehensive summary that reviews all the mentioned agreeing tasks in this section, as well as their respective taxonomies. Taxonomies are set in a yearly ascending order and the Agreeing Tasks are ordered from the most common, along all taxonomies, to the least common, in case of a tie there is an alphabetical tiebreaker. This type of arrangement allows for small observations since it is possible to perceive that tasks including *Relationship* and *Distribution* are prevalent in the data visualization literature, whereas *Discover* and *Spatial Understanding* can be considered more niche tasks.

Agreeing Tasks / Taxonomies	Wehrend et al. 1990	Amar et al. 2005	Valiati et al. 2006	Abela, 2008	Yau, 2011	Few, 2012	Münzner, 2014	Laha et al. 2015	Börner, 2019	Quadri et al. 2021	Li et al. 2023
Comparison / Compare / Nominal Comparison / Differences	X	-	X	X	X	X	X	-	X	X	X
Relationship / Correlate / Relationships / Correlation / Correlations	X	X	-	X	X	X	-	-	X	X	X
Distribution / Characterize Distribution / Distributions / Determine	X	X	X	X	-	X	-	-	X	X	-
Composition/ Part-to-whole / Proportions / Compositions	-	-	-	X	X	X	-	-	X	-	X
Identify / Retrieve Value	-	X	X	-	-	-	X	-	-	X	-
Rank / Order / Sort / Ranking	X	X	-	-	-	X	-	-	X	-	-
Search / Locate	X	-	X	-	-	-	X	X	-	-	-
Cluster	X	X		-	-	-	-	-	-	X	-
Derive / Compute Derived Value	-	X	-	-	-	-	X	-	-	X	-
Determine Range	-	X	-	-	-	-	-	-	-	X	-
Find Anomalies		X		-	-	-	-	-	-	X	-
Find Extremum	-	X	-	-	-	-	-	-	-	X	-
Identify / Explore	X	-	-	-	-	-	X	-	-	-	-
Infer / Discover	-	-	X	-	-	-	X	-	-	-	-
Spatial Understanding / Distribution	-	-	-	-	-	-	-	X	-	-	X

Table 3.1: Overview table of agreeing tasks, their original taxonomies, and respective relationships.

Following the table's order, one of the core tasks is Compare. There is little to deviate from the foundational notion of comparing elements or their characteristic. Thus, all instances of comparing are within the same group. *Compare* [127], *Compare* [116], *Comparison* [2], *Different* [133], *Nominal Comparison* [35], *Compare* [80], *Comparisons* [16], *Compare* [92], and *Comparison* [66]. Some of these tasks are applied to slightly different

contexts, e.g. Few's *Nominal Comparison* focuses on comparing discrete values in a nominal scale. Even so, the variance is rather small.

One of the most popular tasks found, in all taxonomies, was the notion of identifying relationships between elements or concrete attributes: *Correlate* [127], *Correlation* [8], *Relationship* [2], *Relationships* [133], *Correlation* [35], *Correlations/Relationships* [16], *Correlate* [92] e *Relationship* [66].

Distributions were a rather common task in our research. Albeit, two main branches depend on the goal of determining distributions of values or the spatial distribution of data due to their location. The first being the more "traditional" methodology where we can find: *Distributions* [127], *Determine* [116], *Characterize Distribution* [8], *Distribution* [2], *Distribution* [35], *Distributions* [16], and *Characterize Distribution* [92]. Alternatively, branch number two can be seen as the "esoteric" approach with *Spatial Understanding* [61], and *Distribution* [66] that involves a concept based on location and space. According to the authors there exists a clear relationship between the spatial distribution of elements and their attributes' values.

Composition [2], *Proportions* [133], *Part-to-whole* [35], *Compositions* [16], and *Composition* [66]. Although slightly diverse, all the tasks fit with the general idea of representing different parts that contribute to forming a whole.

The tasks of *Retrieve Value* [8], *Identify* [80], *Retrieve Value* [92], and *Identify* [116] have separate routes and contexts but even still the results are comparable, in a sense that both involve identifying specific information from a dataset.

Organizing is a rather useful task in data analysis and data visualization, it is a recurrent task even though they are not completely convergent. *Rank* [127], *Sort* [8], *Ranking* [35], *Order/Rank/Sort* [16], and *Sort* [92]. There are subtle differences between "Rank" and "Order/Sort", the first one orders elements using an ordinal classification whereas the latter involves ordering data based on some ordinal criteria suggested by the user. Because of their high similarity in sharing the central idea of ordering or classifying, both task concepts were kept together as a unique group.

Identifying elements/attributes is a big part of data analysis. These tasks encompass the need to identify or locate concrete information, either about a specific element or characteristics of the element and/or dataset. *Search* [80], *Locate* [116], *Locate* [127], and *Search* [61] achieve the previously stated objective.

Clustering tasks, *Cluster* [127], *Cluster* [8], and *Cluster* [92], determine clusters of elements that resemble each other or share relationships.

Compute Derived Value [8], *Compute Derived Value* [92], and *Derive* [80] create new values based on statistical operations on existing values.

The tasks related to intervals (ranges) appear as *Determine Range* [8], and *Determine Range* [92] can discover the values within a determined bound range.

Find Extremum [8] and *Find Extremum* [92] have a feature of finding the highest and lowest values of an attribute in an interval. And in one identical lens *Find Anomalies* [8] and *Find Anomalies* [92] identify outlier values regarding the other attribute values.

Identify [127] and *Explore* [80]. Sharing a proposition on which the user does not have previous knowledge of what is looking for, so the necessity is uncovering new information.

Infer [116] and *Discover* [80] are difficult tasks to define specifically because of their higher abstraction. Objectively, they try to reason with the discovery of new knowledge that will result in new information, rules, hypotheses, or questions to visualize.

Finally, the last identified compatible tasks are *Spatial Understanding* [61] and *Distribution* [66], which have their focus set on finding/relating the positional distribution of elements.

3.3.2 Found Hierarchies

After analyzing the agreeing tasks, it was possible to identify the existing hierarchies, both in terms of concepts and how wide is their scope of execution. Characterizing tasks guarantees that the superior levels are always broader and more abstract. In this section, hierarchies are organized, and described thoroughly.

Compare hierarchy, Figure 3.1, hoards multiple agreeing tasks in different taxonomies: *Compare* [127, 116, 16, 92]; *Comparison* [2, 66]; *Differences* [133];. The *Compare* main task with its respective agreeing tasks establish *Deviation*, which demonstrates how one or more sets of quantitative values differ from a reference set, and *Nominal Comparison*, that compares discrete quantitative values using a non-ordered categorical scale, as sub-tasks [35].

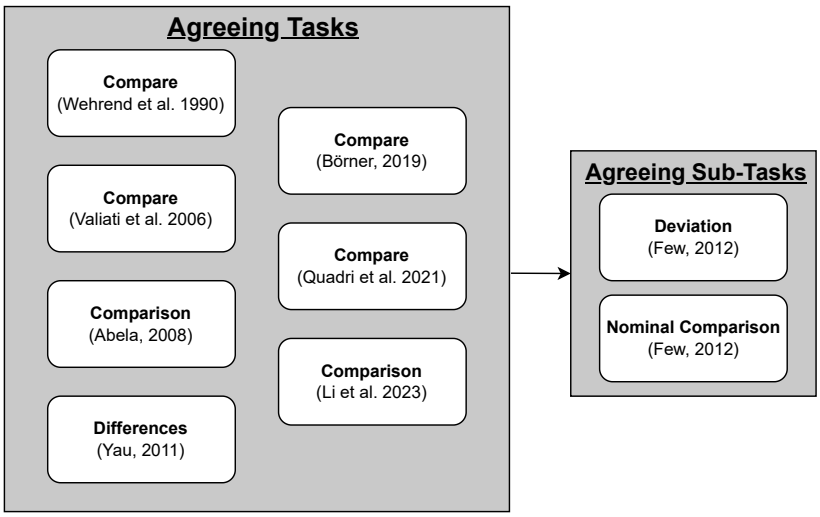

Figure 3.1: Hierarchical diagram of *Compare* agreeing tasks.

The hierarchy initiated by the high-level task *Derive*, proposed by Münzner [80], can be seen in Figure 3.2, produces new datasets, elements, and attributes based on existing elements. This is the highest level possible since it enables the creation of new entries, then comes *Determine* [116] and *Compute Derived Value* [8, 92] in the same lower level. Both create new values based on numerical and statistical operations (e.g. sum, mean, variance, probabilities, etc.).

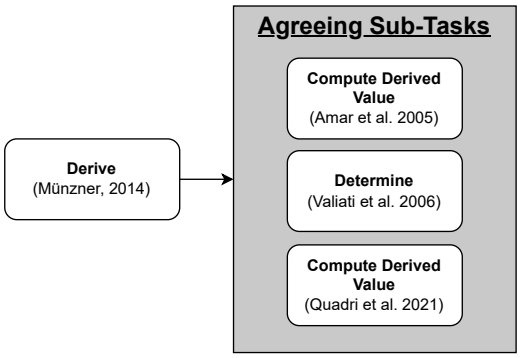

Figure 3.2: Hierarchical tree diagram of *Derive* agreeing tasks.

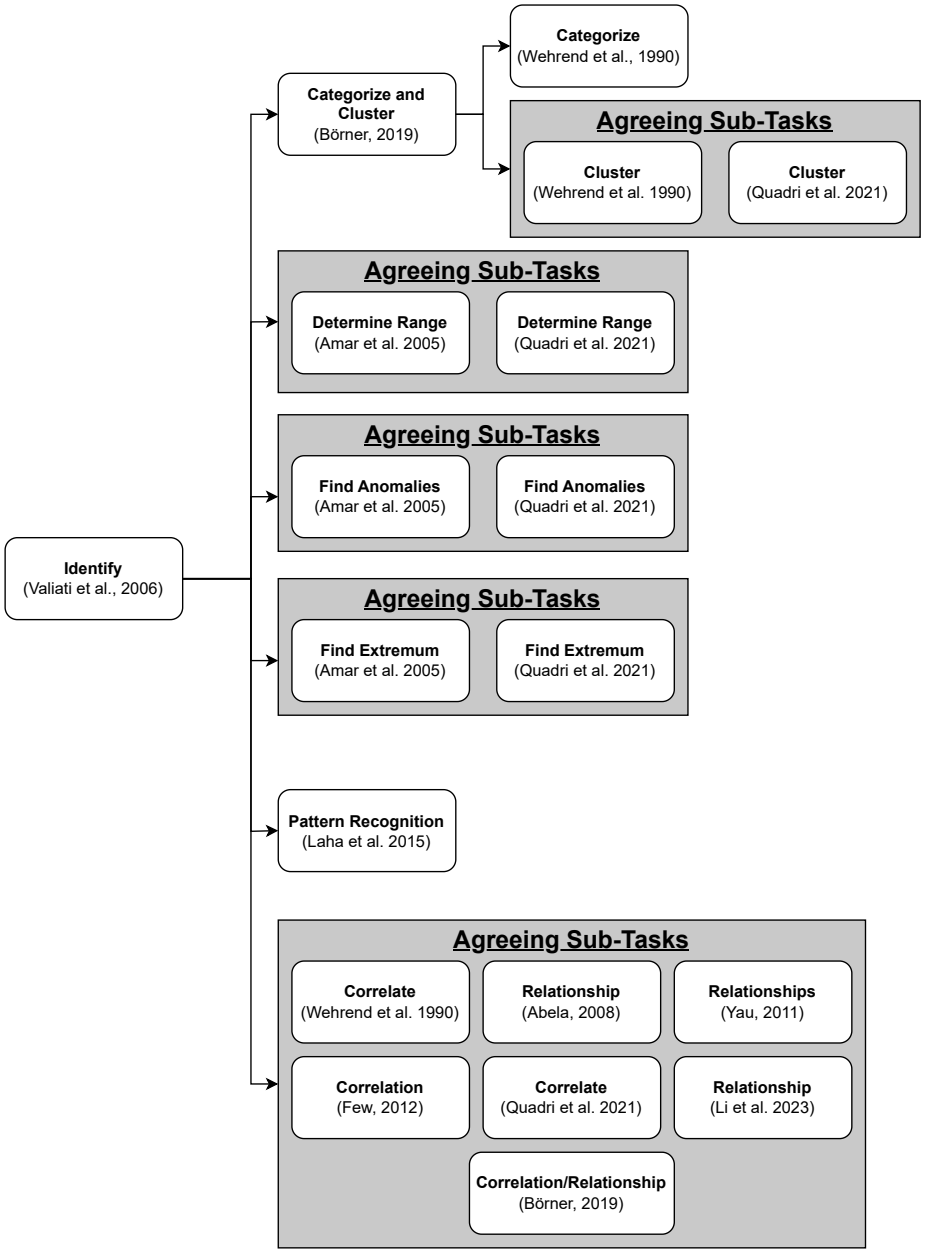

Figure 3.3: Hierarchical tree diagram of *Identify* agreeing tasks.

The biggest hierarchy identified was assembled with the initial task *Identify* of Valiati's [116] given its abstract and far-reaching description. Due to this, we believe that multiple tasks can be understood as sub-tasks of *Identify*. Following the Figure 3.3, which illustrates this exact hierarchy, we consider *Categorize and Cluster* [16], and its respective sub-tasks *Categorize* [16] and *Cluster* [127, 92], as a sub-task of Identify since these tasks involve the identification and discovery of clusters and categories. *Determine Range* [8, 92], *Find Anomalies* [8, 92], and *Find Extreme* [8, 92] are also included as sub-tasks since they have a matching objective of finding a value or set of values, but their specifications change slightly. Identifying and finding patterns is intrinsic to the identity of *Identify*, therefore Laha's *Pattern Recognition* and its abilities to find and signalize pattern is crucial. [61] Lastly, this hierarchy encompasses the ability to find and identify relationships and all the agreeing tasks related to *Relationship* [2, 133, 66], *Correlation* [35, 16], and *Correlate* [127, 92]

It is important to state that tasks that are not included in the **Agreeing Tasks** section, such as the case of *Visualize* [116], *Associate* [127], and *Summarize* [80] present details that are not addressed by other tasks and are not integrated into any hierarchies. *Visualize* is described as a supporting task for analytical tasks. Whereas, *Associate* involves the user creating relationships between objects, something that is not available in any other taxonomy. Finally, *Summarize* has the particularity of presenting an overview of the data, something that is not replicated by any other analyzed task.

3.4 Proposal

As mentioned previously in this dissertation, in section 2.4, a **taxonomy** classifies interconnected concepts through the means of a common subject, this practice of classifying core concepts typically results in a hierarchical schema. Therefore, these crucial elements must be acknowledged to be able to design a domain and data-independent taxonomy, whose main objective is to give access to a set of abstract tasks that are capable of characterizing and answering domain questions submitted by users. The further analysis connected with taxonomies and some of the literature work regarding this domain, made our team identify an assortment of **"generic tasks"** that can be found in different research, papers, and surveys. This results in our proposal, visually demonstrated in Figure 3.4, comprised of five main analytical tasks: **Identify**, **Compare**, **Determine Distribution**, **Organize**, and **Derive**.

Although the investigation work was relevant to our proposal, not all tasks were a priority. One such example is the task **Summarize** mentioned by Münzner [80], it was not integrated into our final taxonomy because of its high-level nature.

It is crucial to emphasize that this proposal is open to integrate new analytical tasks if deemed necessary. Therefore, it can be considered a work in progress.

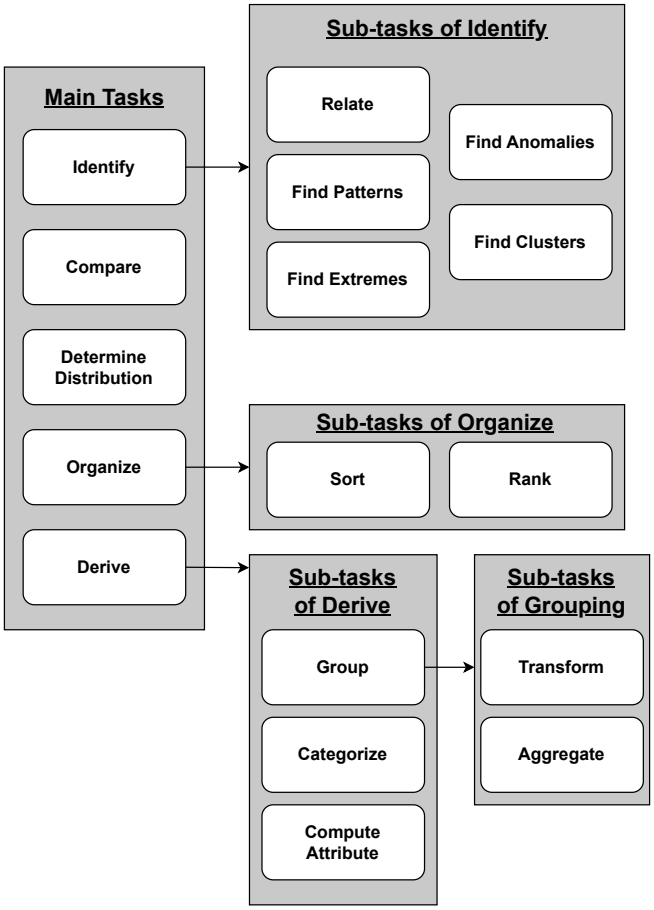

Figure 3.4: Proposed taxonomy's hierarchical tree diagram.

3.5 Tasks

Due to their high-level reasoning, some of these tasks are further decomposed into sub-tasks, to create more understandable and distinguishable characteristics for each task. In this way, we hope to achieve better answers to designated domain questions. Each task will be presented with a description and an example of a possible analytical question.

3.5.1 Identify

Expresses actions closely related to finding/discovering insights within the current dataset. Such a task is commonly found in the visual analytic space, it is referenced in some of formerly surveyed taxonomies. Identify branches into five sub-tasks: **Relate**, **Find Patterns**, **Find Extremes**, **Find Anomalies**, and **Find Clusters**. The hierarchy created is inspired by the result of one of the already analyzed hierarchies. We kept it in our design because of its usefulness in finding multiple types of elements, such as patterns, relationships, extremes, outliers, and clusters.

3.5.1.1 Relate

Responsible for identifying and acknowledging the relationships and interactions between elements with the objective of obtaining data insights. This task was inspired by similarly explored tasks in some of the studied taxonomies, although with different specifications and abstraction levels. Example: Showing how preferences towards a product A are connected, or not, to specific age groups.

3.5.1.2 Find Patterns

Designated patterns can be identified through changes or the lack thereof, within a certain interval. Finding patterns is done by identifying trends in the dataset, as a whole or through multiple smaller intervals. Example: Determine trends of profit, or loss, of the company A.

3.5.1.3 Find Extremes

Solemnly focuses on discovering the values found in the lower and upper echelon values of the dataset. The extremes can be either local or global maximums/minimums. Example: Identify August's highest and lowest daily temperatures.

3.5.1.4 Find Anomalies

Finding anomalies closes into finding the outlier values in the data. Similarly to Find Extremes the outliers can be found locally or globally. Example: What are the regions, in the country A, that have considerably different electrical consumption levels per capita.

3.5.1.5 Find Clusters

Clusters are groups of data elements that share identical or similar attributes. Finding clusters involves identifying groups that share characteristics since clustering and segmenting report additional knowledge about the dataset, mostly in a non-deterministic way. Example: Determine what are the main groups of sold items, knowing each item is categorized.

3.5.2 Compare

Extensively used in data analysis as a result of the innate necessity of comparing data elements, values, and characteristics. The main objective is to classify/quantify the differences in data. Some tasks - Find Extremes, and Find Patterns, for example - rely on comparison in an implicit manner, where it is paramount in the logical steps to achieve the desired tasks' results. Example: Compare the profit of three companies A, B, and C.

3.5.3 Determine Distribution

The concept of distribution is thoroughly used in visual taxonomies and is often seen as an important task. In most of taxonomies, it is related to how values in data, typically in an attribute, are distributed in the whole dataset, highly related to statistical analysis. However, some taxonomies create a distinct method of distribution which is related to how data is distributed position-wise. Our task **Determine Distribution** endorses the regular statistical approach. Example: Observe who are the clients with the most orders realized.

3.5.4 Organize

It comes to fruition as an adaptation of tasks like rank [16, 35, 127, 8], order [16], and sort [16]. Such a task offers both **Rank**, in which a classification system organizes the elements in their respective position in the ranking, and **Sort** which organizes the data based on a user-defined heuristic. Example: Determine the order of students based on their grades.

3.5.5 Derive

Although not directly correlated with the creation of visualizations, the task performs a crucial component in the taxonomy. Regularly the available data does not specify all the necessary information, thus sometimes is necessary to derive attributes, elements, and datasets based on already existing information, supposing a wiser data analysis. Derive presents an essential set of tasks that are exclusive relative to data and precede the display of visualizations. It gets specialized into three sub-tasks: **Group**, **Categorize**, and **Compute Attribute**.

3.5.5.1 Group

Designates groups of elements that must be created by derivation. The groups are deterministic since the user decides the criteria for the group design. Each grouping task contains one operation: Transform, and Aggregate.

- **Transform -** Function that allows the derivation of each group to realize a transformation to every element within the group. Example: Apply a function to all values of data field A.
- **Aggregate -** Function that allows for data processing, in a way that shapes a summary of the elements using a statistical function, like mean, std, count, etc... Example: Calculate the resulting average in a set of values.

3.5.5.2 Categorize

Data categorization can be considered a "manual task" where the user is responsible for labeling the data into categories, even if it does not already exist in the dataset. It is a

more refined version of grouping and clustering. Example: Categorize a set of data that represents the evaluation of products in shop A.

3.5.5.3 Compute Attribute

Allows the creation of new attributes by utilizing already existing data. These attributes are inserted into the rows of elements, or groups of elements, and become part of a new dataset as a result. Example: Create an attribute that represents the percentage of the population in district A, concerning the overall population.

3.6 Translation of Domain Questions to Tasks with LLM

LLM are becoming an increasingly popular and well-established field that is starting to spread into other domains of expertise. Language models are pre-trained models with vast amounts of text data. This results in a highly capable system that can execute natural language processing tasks [32]. These processing capabilities extend to multiple fields within the scope of language processing and there exists a wide range of possible tasks, from a practical perspective [20]. Self-attention is the primary concept that makes this processing possible, an attention mechanism that relates different sequence positions to compute a sequence representation and enables reading comprehension, summarizing, and learning task-independent sentence representations [119].

Our objective was to investigate how this technology could help translate domain questions into analytical tasks due to its natural language processing capabilities. Tests conducted contributed to both evaluating the possibility of having automatic translations performed by an **LLM** - these tests can be found in chapter 6 - and understanding the current state of our taxonomy. It is important to ascertain that the objective was to understand the capacity and limitations of this technology in translating user's domain questions into appropriate analytical tasks, by not needing to rely on human input (e.g. a data engineer).

The first tests performed on **LLM**s helped the team recognize a few rough edges in our proposal that were fixed accordingly. Adding some tasks, e.g. Aggregate, came as a direct consequence of testing with this equipment, since creating a more machine-friendly task selection could result in better solutions.

3.7 Comparative Analysis

As a wrap-up to the current thought process, regarding the formulated proposal based on knowledge extracted from the literature and an iterative testing progress, through the application of said taxonomy in varied domains with examples. The final result is seen as a hierarchy composed of tasks, sub-tasks, and sub-sub-tasks. An evaluation was crucial

to summarize the advancements taken and how the tasks interact differently with each example.

Methodologically speaking, to conclude the evaluation procedure and to consolidate our proposal's integrity, it must leverage a comparative analysis with another taxonomy. Taking this into consideration, the selection relied upon Münzner T. [80] typology, due to its relevance and different approach to data visualization.

Analyze, one of the high-level actions, characterized by a stout interaction component, which is not our focus will be included in this analysis. Thankfully, the actions *Discover* and *Derive* are out of that interaction scope and can be used to evaluate our proposal. The first action reflects the uncovering of new knowledge, that did not exist previously, around observing the data. This is something that our task **Identify** can have as a direct consequence attributed to the results of its sub-tasks. Since most of those sub-tasks are related to finding and/or discovering information, then we can assume a link to Münzner's action. Regarding the second task, our proposal synthesizes a **Derive** task and its respective sub-tasks which invoke similar capabilities.

Search, as an action, has the objective of interesting data for users, based on the user's knowledge, or lack thereof, of the element location or characteristics. The task has four possible executions: *Lookup*, *Browse*, *Locate*, and *Explore*. We do not have corresponding tasks because our model has access to the element attributes and consequently they can be identified. There is also the possibility of filtering the data with user-determined heuristics and acquiring similar results.

Lastly, ***Query*** allows three functions to be applied to data: *Identify*, responsible for retrieving information (e.g. attribute values) of an element. In our proposal, there is no possibility of gathering a single value although we can achieve some resembling results; ***Compare***, allows for the comparison of elements. We also have a directly corresponding task named **Compare**; *Summarize* is a rather peculiar task for which we do not provide an alternative, especially because of its abstract properties and the overview of the data.

3.8 Task's Visualization

In our proposal, tasks express themselves unto data as a means to an end. That end is to perform correct/adequate visualizations for human consumption. Conceptually, our approach to achieve said visualizations is to mix extracted knowledge, from the available literature, and knowledge from experts in our team. Besides the multiple studies that focus on different types of graphs, the presence of "visual encoding"-specific papers allows the complement of both research spheres, to attain the best possible visualizations. Although these investigative works have quantifiable metrics and well-constructed tests, data visualization is still not a "one size fits all" kind of field. The existence of almost infinite domains that have different approaches, domain questions, data types, and ways to be visualized contributes to this notion.

For example, in datasets with spatial data, like coordinates (latitude and longitude), the presence of a real-world map in which the data can be accurately located is important to properly aid the viewer in situating himself visually. This happens since we are used to analyzing maps but not analyzing coordinates, without a visual support of a map. The diversity of knowledge and preferences for each user also contributes to an added difficulty, e.g. for the same chart Person **A** might prefer a scatterplot while Person **B** would lean more towards a bar chart. this makes it statistically unfathomable to draw consistently the best visualization to each individual [21].

Addressing our taxonomy, the following tasks present the respective preferences:

- **Relate**: A study classifying visualizations for correlation tasks [46] determined scatterplots as the graphs that feature the best accuracy. Saket et al. conducted another visualization evaluation study [100] and concluded that for a small number of entries, participants [users] obtain the best results when analyzing line graphs. Some studies determine the best visualizations by the application of statistical models [99, 53, 46] that show a trend for scatterplots and parallel coordinates, for positive and negative slopes respectively. Line graphs and bar charts tend to have positive accuracy and temporal results. When speaking of encodings for this task, the most expressive are position and size.

- **Find Patterns**: Some studies tackle heat maps, line charts, area charts, and vertical bar charts as the main adequate visualization options for pattern identification and recognition [2, 45]. Heat maps are capable of showing patterns and variations between multiple variables due to their use of their typical color and size encodings. Line charts tend to augment the sense of time perception through the combination of a numbered axis with different marks. Area charts emphasize long-term tendencies and finally bar charts present specific patterns for individual values [35].

- **Find Extremes**: Scatterplots, bar charts, and line charts are valid options to check for extremes [100, 103, 92], either locally or globally. In the same line of thought, multiple papers address via a comparative analysis the good performance of the previously mentioned visualizations [100, 87]. In a bar chart, the bars convey an easy way to detect maximums and minimums. Whereas, line charts use their slope variation to find values. Usage of size and shape, as visual encodings, is crucial to have a straightforward response [65].

- **Find Anomalies**: Outlier values are better found in line charts and scatterplots via a pre-attentive process, such as color, size, shape, or orientation [92]. Very similar to "Find Extreme" this task can benefit massively from usage of pre-attentive elements [65, 44].

- **Find Clusters**: Clusters as data groups are primarily identified in scatterplots, and bubble charts [92]. The grouping and utilization of multiple encodings to visually

enhance the groups make it easier to find these elements.

- **Compare**: The comparison of values is generally preferred, in terms of response time and accuracy, when combined with bar charts and/or line charts. For bar charts, there is a specific contribution from side-by-side bar charts which help identify differences between attributes' values. While line charts allow for multiple displays of lines, as well as multiple types of encodings. Empirical results demonstrate the utility of line graphs when comparing a low or medium number of entries and for an adequate graph axis size [92, 40]. Size, color, and shape are the most expressive encodings for this task to facilitate the discovery of intrinsic differences.

- **Determine Distribution**: To facilitate the visualization of distributions, graphs such as kernel density estimation, line charts, bar charts, histograms, and Gantt charts are all possible options. Scatterplots use position, size, and color to maximize the perception of the distinction between values [113]. Bar charts and histograms are generally recognizable when envisioning, for example, normal, exponential, and poison distributions.

- **Organize & Derive**: Due to these tasks' functionalities, which are mainly related to modifying or deriving data, they do not present the need to generate visualizations. This means that they can be understood as non-visualization tasks, or as intermediate analytical tasks. As a result, although useful for data analysis, these tasks contribute to pre-visualization, through filtering and processing of data, or in post-visualization re-organizing displayed data.

4

Modeling

4.1 Introduction

This chapter will proceed toward defining and demonstrating the core focuses and multiple concepts seen throughout the development of the foundational model-driven approach meta-model, as well as the development of a small data repository.

4.2 Conceptual Modelling & Meta-Model

The created meta-model was defined as a supportive model capable of characterizing other conceptual models, user requirements, and data visualizations within a specific domain. When analyzing the model, which is separated into three main components, the focus jumps primarily to designate the following modules:

- (1) **Data Module** - The conceptual model that is being analyzed must have its data completely defined. This module has all the necessary notions that allow for an easier interpretation of the presented dataset. A dataset is composed of items, that can be thought of as single entries in the dataset, these entries are split further into attributes that need to be categorized by their respective data types.
- (2) **Analytical Module** - A more intricate module considering the mapping from user requirements to data is made here. Domain questions are broken down into a set of analytical tasks that are able to build an answer to said questions. Tasks result in attributes, items, or datasets, which are called the tasks' targets.
- (3) **Visualization Module** - Visualizations must show the viewer how to interact with knowledge. The result is the retrieval of information from the visualization itself. Visualizations are constituted by charts and chart components, e.g. headers, axes, and visual marks, which are directly connected to available data fields, that design the visualizations.

Münzner's typology [80] divided into "Why?", "What?", and "How?", heavily influenced our model during its development. This influence translates into the meta-model, where a

module is represented by a question: "Why?" is the **analytical module**, "What?" describes the **data module**, and "How?" is represented by the **visualization module**.

Looking more in-depth into the model, all three conceptual modules are interconnected and relate heavily to one another. Setting these relationships enables a panoramic view of the meta-model, which can be seen in the annex IV, where the symbiosis is tangible and conceptually well-defined. By analyzing individual model modules, the frame of mind becomes the following:

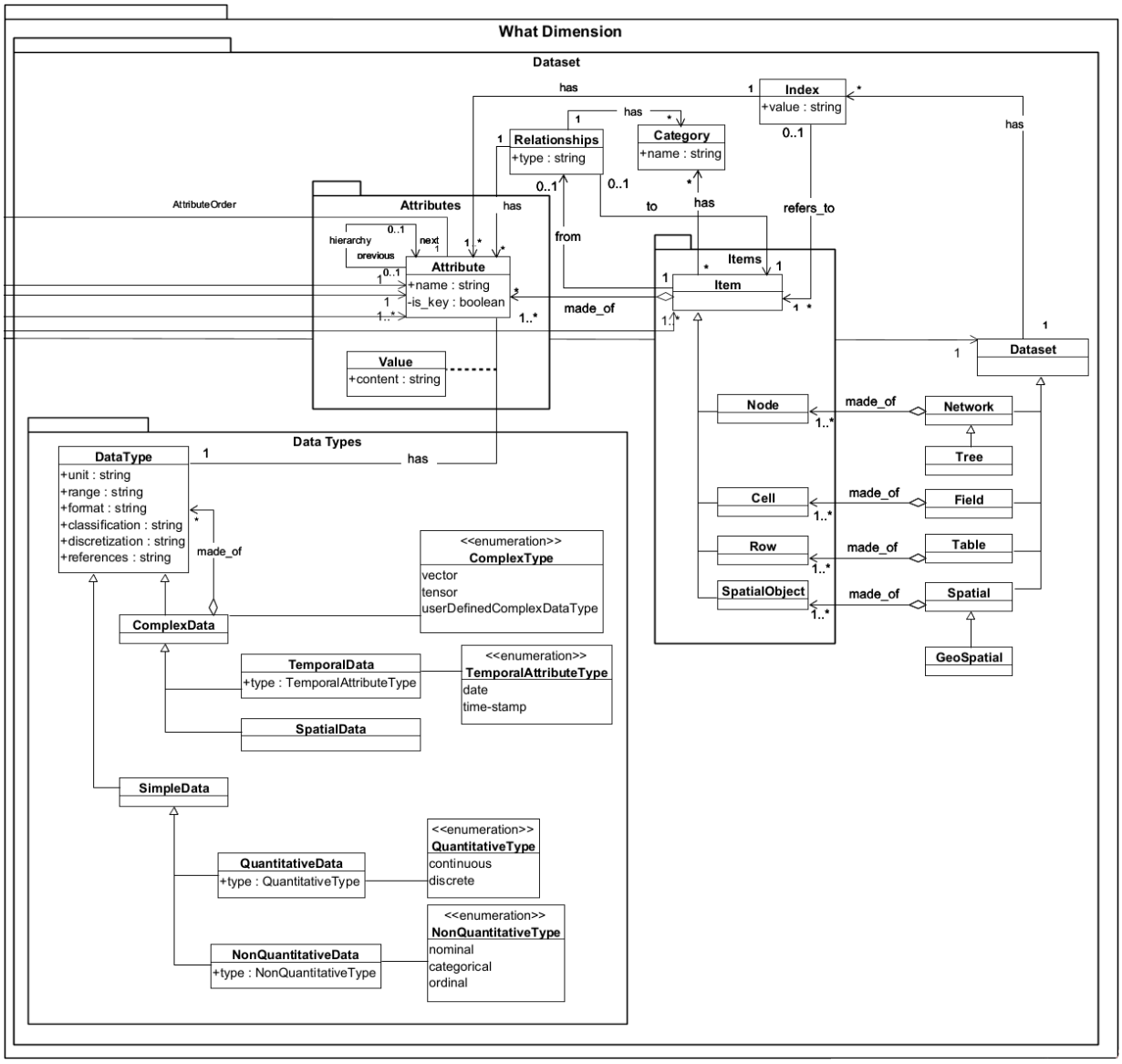

Figure 4.1: Meta-model's unattached **Data Module**.

Data Module, exhibited in Figure 4.1, is responsible for defining and labeling the existing data. Initially, the dataset must be specified by the different available data structures, and these data structures are later broken into item structures. Currently, these are the data structures and item structures at the model's disposal: GeoSpatial, a specialization of Spatial structure made of SpatialObjects; Table, constituted by Rows;

Field, construed by one or more Cells; Trees, which are a specific type of Network, are data structures that utilize Nodes as atomic entities. Items, and respective item structures, are comprehended of one or multiple Attributes, they can be labeled into Categories, and organize Relationships between themselves [Items], their categories, or their attributes.

Attributes are a blend of the attribute's name and value, each pair of these properties is an item entry, and a set order that expresses their importance in the analytical task's targets. The attribute values are characterized by a Data Type. A data type class is categorized as either Complex Data - it encompasses vector, tensor, or user-defined types, that are specialized into Temporal (i.e. complete date, instant time-stamp) or Spatial Data - or simple data that is designated as Quantitative (i.e. continuous, discrete) or Non-Quantitative (i.e. nominal, categorical, and ordinal).

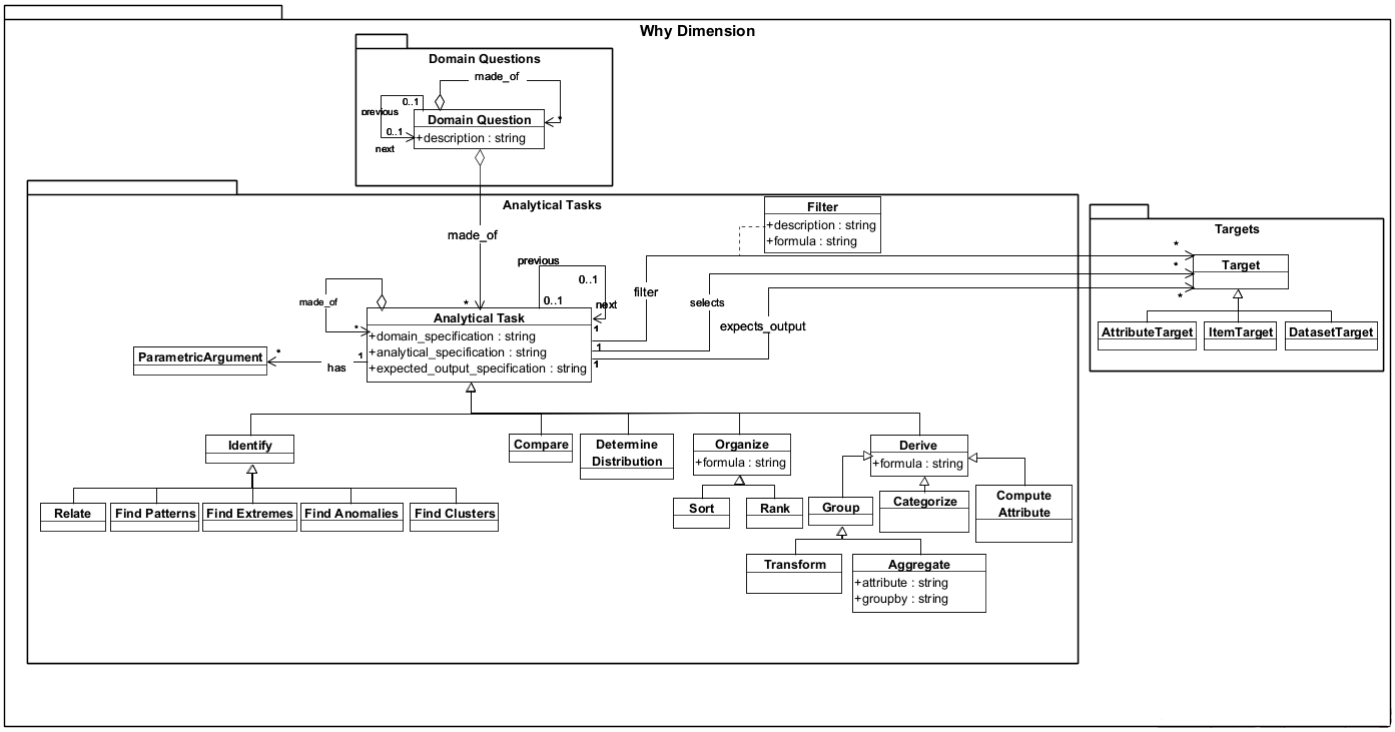

Figure 4.2: Meta-model's unattached **Analytical Module**.

Analytical Module, present in Figure 4.2, is initially composed of domain questions, defined by the users, that get to be deconstructed into analytical tasks. When outputted, such tasks are transformed into attributes, items, or datasets, depending on the expected output. These tasks also hold information relative to the domain and the analytical specification. They establish orders between themselves, have properties regarding what target data is inputted, what are the data filtering requirements, and ultimately the expected output.

Existing analytical tasks, expressed in the model, result from the conducted research on the literature's taxonomies, that served as the basis for the already discussed proposed taxonomy. Similarly to the taxonomy proposal, the tasks are organized in a way that mirrors the taxonomy's hierarchy. Conceptually, tasks differ in abstraction level and objectiveness.

Though the differences between visualization-creating tasks and data-processing tasks are not stated in the model, they are communicated via the model constraints.

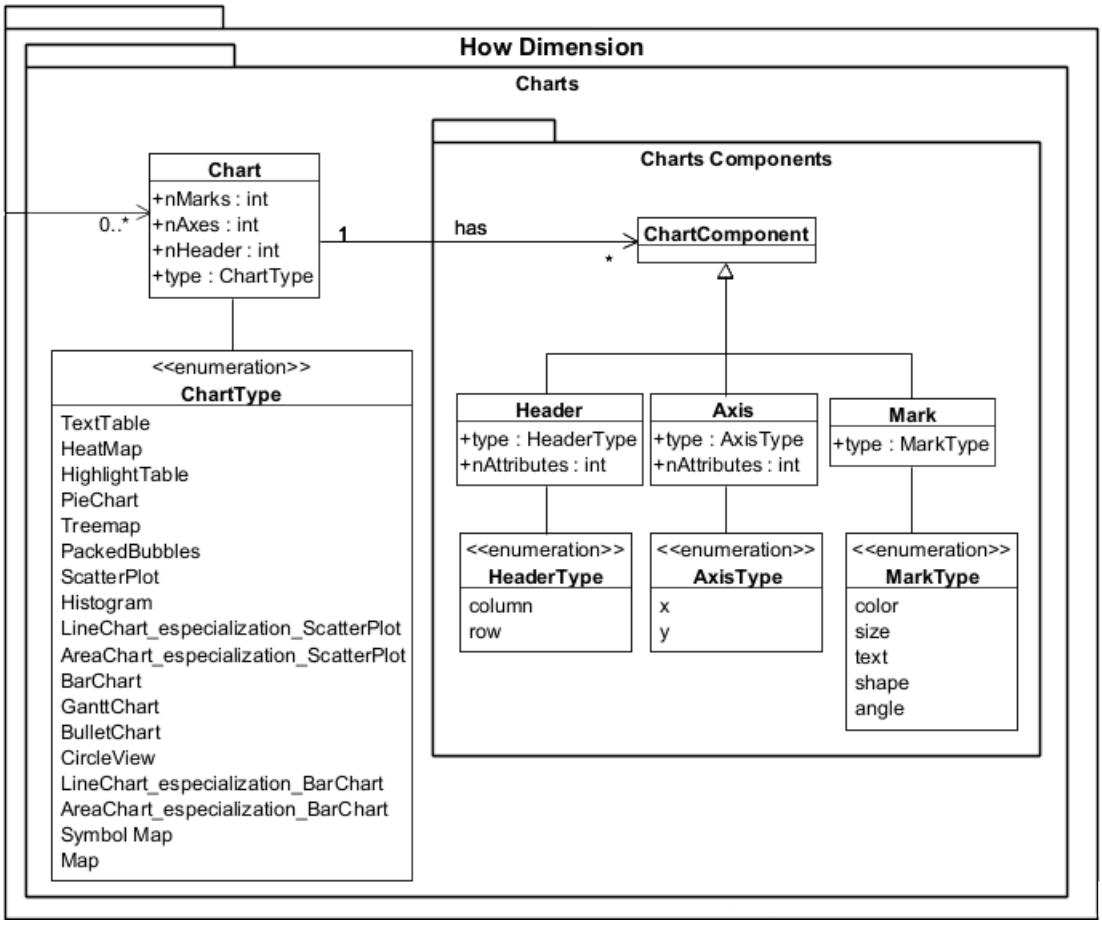

Figure 4.3: Meta-model's unattached **Visualization Module**.

The third and last component, displayed in Figure 4.3, **Visualization Module** depicts and formalizes an amalgamation of facts that result in an adequate visualization, capable of answering the initial domain question utilizing the available data. Visualizations are composed of Charts and their respective Chart Components that map data attributes, originating from the **Data Module**, to one of three possible specializations that define the visualization design choices: Headers (column, row), Axis (X, Y), and Visual Marks (color, size, text, shape, and angle). Alongside the resulting analytical task, from the **Analytical Module**, it is possible to establish the eligible types of ChartType and some of its design choices.

As it can be observed in Listings 4.1,4.2, and 4.3, the conceptual model follows an array of constraints that are crucial to keeping up consistency in the **Visualization Module**. These constraints were written using Object Constraint Language (OCL) [83], a UML rule-based language for constraint definition within models and meta-models, which offers an extra layer of abstraction to the model capable of solving inconsistencies or ambiguities that the model might not be able to express. In this meta-model, they act upon

information regarding Charts and Chart types focusing on chart instantiation (minimum) requirements, feasible visualizations per task, and consistency invariables. The examples below reflect one constraint for each focus and merely show individual examples.

In the first **listing 4.1**, the **constraint details the minimum requirements** for the respective Charts: Scatterplot, Histogram, LineChart, AreaChart, Map, and SymbolMap. These charts must have two axes, no headers, and at least two marks to be instantiated and drawn on the visualization space.

```
Context Chart
inv: (self.nAxis = 2 and self.nHeader = 0 and self.Mark >= 2)
implies
(self.type = 'Scatterplot' or self.type = 'Histogram' or
self.type = 'LineChart_especialization_Scatterplot' or
self.type = 'AreaChart_especialization_Scatterplot' or
self.type = 'Map' or
self.type = 'SymbolMap')
```

Listing 4.1: Example of the minimum requirements constraint for charts.

Listing 4.2, constructs an **invariant that prohibits the existence of a Chart for the analytical task Derive**. This is to say that the Derive task is incapable of generating visualizations.

```
Context Derive
inv: self.AnalyticalTask.Chart.isEmpty()
```

Listing 4.2: Example of a consistency invariable constraint.

The **listing 4.3** allows for the **reasoning of feasible/adequate visualizations based on a specifically selected analytical task**. For this example, it reads that if there is an analytical task **Relate** then the available charts must be one of the listed ones.

```
Context Relate
inv: self.Identify.AnalyticalTask.Chart.notEmpty() implies
(self.Identify.AnalyticalTask.Chart.type = 'Scatterplot' or
self.Identify.AnalyticalTask.Chart.type = 'LineChart' or
self.Identify.AnalyticalTask.Chart.type = 'BarChart' or
self.Identify.AnalyticalTask.Chart.type = 'HeatMap' or
self.Identify.AnalyticalTask.Chart.type = 'HighlightTable' or
self.Identify.AnalyticalTask.Chart.type = 'Map' or
self.Identify.AnalyticalTask.Chart.type = 'SymbolMap')
```

Listing 4.3: Example of feasible visualizations constraint.

The meta-model was further classified using a research reference framework, as seen in Table 4.1, and 4.2. Contextually, the model design establishes four crucial categories.

This sets the goals and links them to related work in the field of conceptual modeling research, this is considered the first part of the model's characterization.

Intended Purpose was chosen from a set of existing options as knowledge representation since one of the targets of this dissertation is precisely the representation of knowledge by the nature of visualizations. The **Intended Domain** of the conceptual model is data visualization since it is one of the model's outputs. Setting the **Domain of case study/Examples** was the characterization of data and instantiation of charts. Concluding, the last category **Intended Users** was agreed upon for both the general public and data analysts.

Intended Purpose	Knowledge Representation
Intended Domain	Data Visualization
Domain of Case Study/Examples	Instantiation of charts
Intended Users	General Public, Data Analysts

Table 4.1: Context characterization, according to Delcambre et al. framework [29], of our meta-model.

The second segment of model classification is closer to the contribution of our model and the contribution from other conceptual models expressed within our model, either through direct comparison with existing models or by indicating formalized modeling languages or patterns.

In this table, 4.2, the **Conceptual Model** is evaluated as a conceptual model for data visualization creation which formalizes a **Conceptual Model Language** for data processing and visualization.

	Contribute or Extend	**Evaluate or Compare**	**Formalize**	**Philosophical Grounding**
Conceptual Model(s)		New conceptual model for instantiation of visualizations		
Conceptual Model pattern(s) meta-model(s)				
Conceptual Modeling language(s)			Data Processing & Data Visualization	

Table 4.2: Research characterization, according to Delcambre et al. framework [29], of meta-model.

As a summary, the meta-model was created with the premise of following diligently the model-driven analytics approach, initially demonstrated in Figure 1.1. By following the methodological approach, there is a parallel relation to be set between the approach components and the meta-model modules: **Data Module** is the representation of **Domain and Data Mapping**; **Analytical Module** constitutes the approach's **Analytical Tasks Identification** and the **Analytical Task Processing**; Lastly, the model's **Visualization Module** accounts for the **Visualization Processing** component.

This overview shows that the model fits symbiotically with the delineated hypothesis approach. It can incorporate data analysis and processing, translation of user requirements

into designated analytical tasks, processing of tasks, and generation of visualizations.

4.3 Data Repository

The necessity of keeping the data organized for further analysis and testing came as a natural evolution after wrapping up the meta-model. A case in point for the repository idea was keeping all data and model outputs in a single knowledge repository, with a particular organized hierarchy. One of its primary objectives is attaining consistency and creating an environment where retrieval for processing or analysis of individual examples is simplified through the following concepts:

- **Domains** - Primary concept that allows for establishing the examples, called use cases, and their respective contexts. From here, the different examples sprout and branch into the data organization, data mapping, and domain questions.
- **Use Cases** - Represent individual examples with direct access to the dataset, to ultimately obtain results that clearly answer to a particular use case.
- **Datasets** - Holds the dataset files of a specific domain. Typically in one of the most common extensions available, like comma-separated values or JSON files.
- **Domain Questions** - Initially and iteratively appear throughout the example analysis as natural language representations that must be translated and assigned into analytical tasks.
- **Domain Mappings** - Maps the conceptual model classes into the domain data available in the dataset.
- **Analytical Tasks** - Collections of analytical tasks identify the necessary "steps" to answer a particular domain question. The execution of these tasks results in an adequate answer (output).
- **Results** - The reflection of the model output, has information regarding the evaluation of the analytical tasks and resulting targets.

Analyzing the conceptual model, available in figure 4.4, it is composed of a tree hierarchy that represents the system's internal organization. A system is made of Domains, and specific domains contain individual examples within the same context. Domains can accommodate multiple Use Cases, which house all sole examples. Corresponding examples are further cataloged with the respective dataset, data mapping, and users' domain questions. Data is kept in the dataset file(s), data mapping discloses the connections between the data from the conceptual model to the data present in the data, and finally, domain questions lay the groundwork basics to be dissected into adequate analytical tasks.

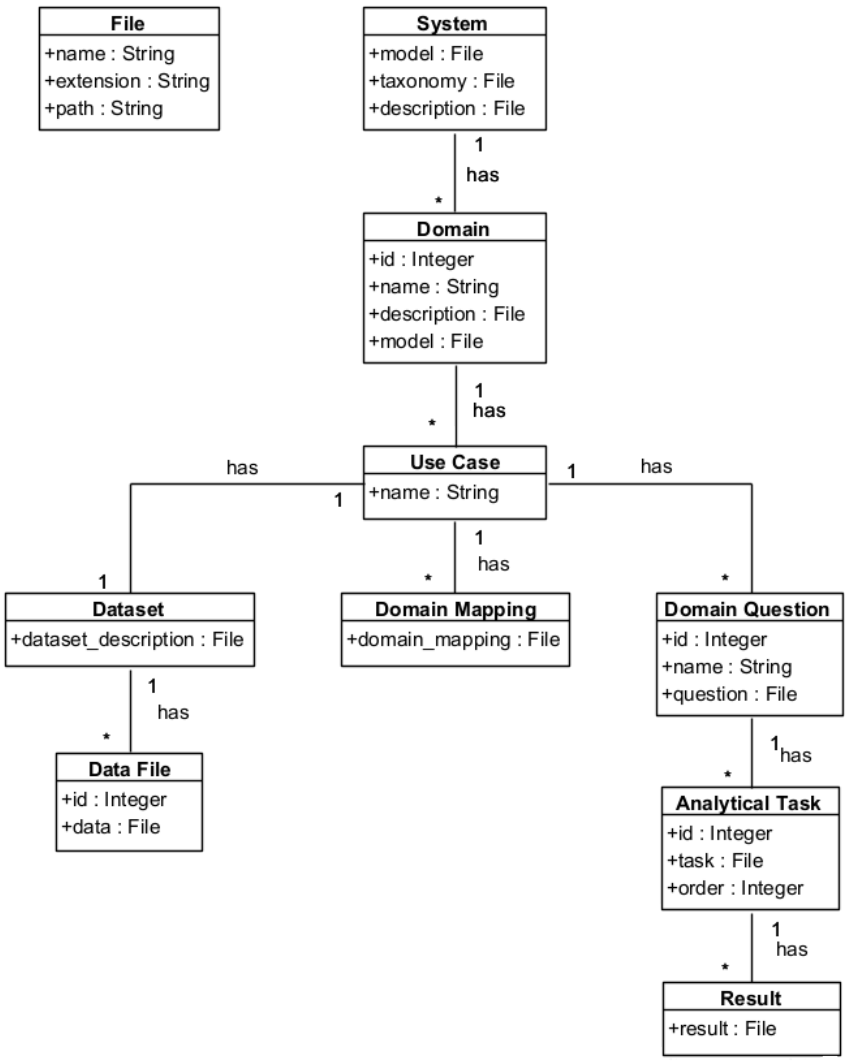

Figure 4.4: **Data Repository**'s Conceptual Model, taken from Almeida A. [6].

The repository became an important support tool for instantiating examples to evaluate and assure that the meta-model follows the expected approach, which was predominantly tested by Almeida A. in her dissertation [6]. The implementation and testing of an automatic tool able to fetch information from the repository and provide the information to a visualization system, that is able to generate visualizations based on such information (data, analytical tasks, and the meta-model output) will be executed by myself.

5

Recommendation System Analysis

This chapter delves into recommendation systems, more specifically the systems that were addressed in section 2.6.3, and the selection of an individual system to be integrated into this dissertation. Apart from this choice, which will be made by following a set of guidelines, the expansion and evolution of the recommendation system will also be a topic of discussion in this chapter. It was important to find a compromise between existing, working, up-to-date, and feasible systems to our approach hypothesis. Our choice was simultaneously done through the lens of aligning the requirements of this dissertation's context to facilitate system inclusion.

Nevertheless, the remainder of this chapter dabbles into partly implementing our proposed taxonomy, executing system training, and creating a parser that supports automation and execution of additional analytical tasks.

5.1 Choosing a Recommendation System

In an effort to choose, and subsequently integrate a recommendation system that would be deemed suitable enough for this project, some guidelines were delineated in terms of available features that such a system should have. Creating such criteria ensured that the chosen system would meet our requirements and also demonstrate flexibility to blend into our methodology. The previously stated guidelines were determined to be the following:

- (1) **Integration & Extensibility :** The chosen system must be capable of being extended, either through creating abstractions or by directly being able to add new features. This stems from the fact that our expectations are the possibility of structuring the proposed taxonomy within the model itself, approximating it to the project's vision, and allowing for translated domain questions to be taken into consideration.
- (2) **Completion :** The system must provide, as output, fully-fledged visualizations and not only visual encodings or visualization rankings. From this, the system must be built upon a library that enables the creation of multiple visualizations, independently of their complexity.

- (3) **Efficiency and Lightweight-ness :** The model should, in theory, be efficient, and have minimal bloat. While simultaneously being suited to display results provided rapidly, Keeping the core data visualization concepts without detriment to the system.
- (4) **Easy Instantiation of Visualizations :** Work behind the visualization request must be kept simple, even if it works with a wide range of information. Facilitating this front allows us to create an automated program that combines the data information and the requirements through the usage of our data repository.
- (5) **Training & Existing Dataset :** Due to the presence of machine-learning-based models, it would be incredibly hard to create a dataset that could resonate with good visualizations. Most of these models rely on considerable quantities of learning, training, and testing data. Typically, the quality of the final output is conditioned by the quality and quantity of the initial data. Regarding these datasets, they are arduous and time-consuming if meant to be built from scratch, which would be our case in this specific occurrence.

Considering the constructed guidelines, the pros and cons of each model, and the individual sample testing results, there was a decision to take Draco [132] as the primary choice. Even though it would require the system to be extended and trained, the necessary dataset could be much smaller because **knowledge-driven models can (more) easily include our expert's preferences and literature knowledge**, and implement our taxonomy proposal. Contrasting with data-driven models that need considerable amounts of data and typically do not allow the implementation of knowledge through rules and constraints.

The Draco system utilizes format-specific files with information and details regarding data and visualization properties, this file is parsed and validated through Clingo [38], an ASP system. After that, the specification is translated to a format capable of being interpreted by Vega-Altair [117], and ultimately the visualizations are obtained by the translated and validated Vega-Lite [104] specification Altair generates. This pipeline process is summarized in Figure 5.1.

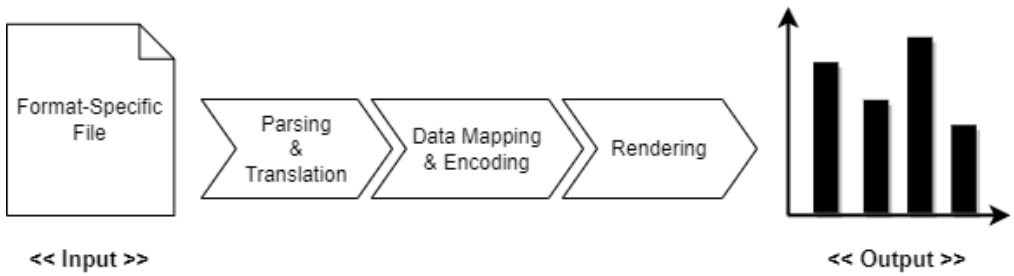

Figure 5.1: Draco's pipeline process, starting on the input file until output visualization.

In actuality, the full pipeline is a bit more convoluted since two components are missing from the previous explanation. Within the **Rendering** step, in Figure 5.1, where Altair

[117] sends his translated specifications to Lite, there is a stack of systems/frameworks that allow for the visualizations to be created and displayed. The higher-level component is Vega-Altair, which works as an API capable of producing stable and valid Vega-Lite [104] grammar specifications, as JSON files. Vega-Lite, the first instance of a present visualization grammar, compiles its specifications files and assembles them with additional expressive information necessary for Vega, the foundational grammar. Vega generates lower-level JSON files, which are typically more dense and complex and unable to assimilate Vega-Lite's high-level abstractions. Vega can be thought of as a visualization grammar closely related to the original Grammar of Graphics [129], allowing for more control and versatility over the visualization's design choices. Finally, Vega renders the specifications into visualizations outputted through JavaScript. This rendering is possible due to Vega utilizing a popular visualization library called D3, which besides the normal visualizations also allows for interactive ones. An interpretation of the Vega stack is illustrated in Figure 5.2

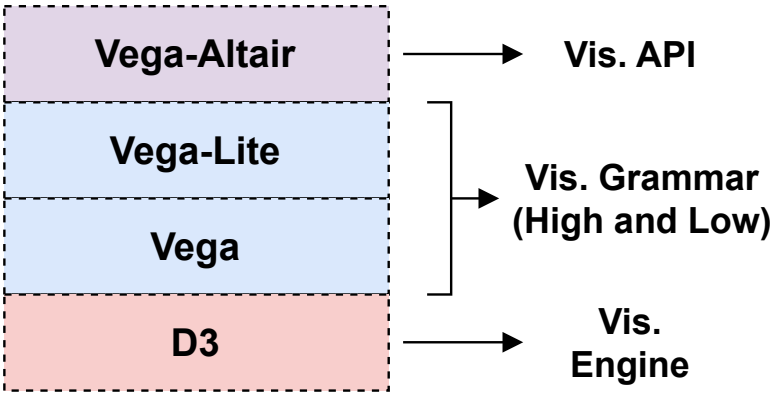

Figure 5.2: Vega's stack framework, inspired by Lees E. image [62].

5.2 System Expansion

Proceeding forward with the choice of Draco as the recommendation visualization system for this project's development, it is important to understand within the system itself how the objectives will be accomplished.

Firstly, it should be reinstated what are the crucial objectives for the visualization system:

- (1) **Inclusion of the proposed taxonomy :** Imperatively speaking, the number one priority is successfully integrating our taxonomy's proposal, as seen previously in section 3.4. It consists of tasks that create visualizations (main focus) and tasks that target data, either for processing or transforming. Projecting these functions correctly into the system and the visualization space is paramount.

- (2) **Achieve adequate results :** Together with the experts on this project's team, it was possible to determine, along with information found throughout literature about visual preferences on charts and encodings, what were considered to be good visualization results for different types of analytical tasks. This becomes increasingly important when it comes to clearly understanding what the domain question is and how the system can adequately answer it.

- (3) **Automation of visualizations :** Developing a program capable of utilizing data and parsing the necessary information to the expected format that will be utilized as an automated input to the system, fundamentally creating a pipeline between the data repository and the final visualization.

Looking at these three objectives, it is clear that two of them [**(1)** and **(2)**] are directly involved with the system and how the system performs, while the third goal [**(3)**] is indirectly connected with what is happening "under the hood". Hence, the first two objectives require changes to the core building blocks of the visualization system, while in contrast, the third goal interacts with the system separately.

5.2.1 Inclusion of the proposed taxonomy

Addressing the **(1)** goal is resolved mostly by virtue of considering the features of the system and how to mesh them with new information and concepts. Draco in its standard version characterizes visualizations by using premises, or facts as the authors call them, that are solved by Clingo (ASP Solver) [38]. Visualizations are achieved by applying different functions, such as **entity**, which is used to describe associations that reflect the creation of objects, and **attribute**, which maps properties (e.g. marks) to separate entities.

Analyzing the system gives further insight into the objects and properties that are authorized to be used in entity and attribute functions, respectively:

- **Field :** Is a representation of the data fields and their properties, such as name, type, unique values, min/max values, and frequency of values.

- **View & Facet :** Views work as grouping agents of objects like marks and scales. Facets are visual operators that allow for the creation of x-axis or y-axis facets for a specific field.

- **Mark :** Graphical mark of the visualization for a View, it draws inspiration from Grammar of Graphics [129]. Available marks are point, bar, line, area, text, tick, and rect. Marks properties are defined by encoding objects.

- **Encoding :** Characterizes data fields and how they should be mapped to visual channels, it is also responsible for defining data aggregations and binned data.

- **Scale :** Scales define the data domain and how it should be drawn in the visualization.

- **Task :** Abstracts over the data visualization concept of analytical task and adapts each task result in an attempt to find the most adequate visualizations for a specific set of tasks.

Considering the already existing object, task, the next step in development is introducing the proposed taxonomy task names into the ASP dictionary and the declarative grammar. Subsequently, this lays the basic foundations from which the constraints, that will later be defined, will exert their preferences regarding charts, data types, and encodings. Data types within the system were reorganized/created to express more closely the data types present in the meta-model: **Quantitative**, **Nominal**, **Ordinal**, **Categorical**, and **Temporal** (Continuous/Discrete).

The system currently accommodates three of our visualization-creating tasks. Since this study/survey of recommendation systems was conducted with the intent to make a proof of concept with a working adaption of our proposed taxonomy. At this stage the system is capable of recognizing **Relate** (subsection 3.5.1.1), **Compare** (subsection 3.5.2), and **Determine Distribution** (subsection 3.5.3). As a consequence of late changes to the taxonomy, the recommendation system utilizes the old name of the **Relate** analytical task, previously named **Relationship**.

Adjacent in Listing 5.1, **there is the assessing of an instantiation example** that offers insight into how data, from the dataset, and properties, for the visualizations, are determined and how these facts influence the system. The example applies to an instantiation file that wants to display the average *discount* per month of the year (*monthyear*). To achieve a sufficiently good answer to this example the analytical task chosen was **Relationship** (Relate). From top to bottom, the listing is composed of:

- **(1)** - The task is selected and decided by the respective adequate analytical task, in this case it is **Relationship**, then the number of rows in the dataset is also annotated. Both are characterized by the function **Attribute**, within the system.

- **(2)** - Defines the data field *discount* and some of its characteristics, such as its data type - in this field section only string, number, and datetime are available types - its inner-system entity name (f,0), and the number of unique values (12). Initially, it is necessary to create the **Entity** (f,0) as a field within the system. The field name and remaining characteristics (type and unique values) are added to the entity by using **Attribute**.

- **(3)** - Very similar to **(2)**, but it defines the data field *monthyear* that is a datetime field.

- **(4)** - Creates the entity **View**, (v,0), where the visualization will be displayed and the **Mark** entity, (m,0), that associates with the aforementioned **View** (v,0) .

- **(5)** - Sets the **Encoding** entities, (e,0) and (e,1), that will be linked to the fields *monthyear* and *discount* respectively. Besides the fields, the encoding also maps these

fields into the axis channels, x, and y. In this specific case, since there is a need to know the average of a data field the file also specifies the **Aggregate** encoding of *discount* grouped by *monthyear*.

- **(6)** - The last entity to be determined is the **Scale**, which is tied to the **View** of the visualization space, and defines the scale data type for the axes and/or encodings, (i.e. ordinal, linear, categorical, and nominal).

```
attribute(task,root,relationship).                (1)
attribute(number_rows,root,9994).

entity(field,root,(f,0)).                         (2)
attribute((field,name),(f,0),discount).
attribute((field,type),(f,0),number).
attribute((field,unique),(f,0),12).

entity(field,root,(f,1)).                         (3)
attribute((field,name),(f,1),monthyear).
attribute((field,type),(f,1),datetime).
attribute((field,unique),(f,1),12).

entity(view,root,(v,0)).                          (4)
entity(mark,(v,0),(m,0)).

entity(encoding,(m,0),(e,0)).                     (5)
entity(encoding,(m,0),(e,1)).
attribute((encoding,field),(e,0),monthyear).
attribute((encoding,channel),(e,0),x).
attribute((encoding,field),(e,1),discount).
attribute((encoding,aggregate),(e,1),mean).
attribute((encoding,channel),(e,1),y).

entity(scale,(v,0),(s,0)).                        (6)
attribute((scale,channel),(s,0),x).
attribute((scale,type),(s,0),ordinal).
entity(scale,(v,0),(s,1)).
attribute((scale,channel),(s,1),y).
attribute((scale,type),(s,1),linear).
```

Listing 5.1: Example of an input file for a visualization instantiation.

After instantiating the new tasks in the model, then it is necessary to set respective constraints to designate the task's preferences and diminish inherent biases lodged in the recommendation system. Constraints have two types: soft and hard.

Soft constraints are present in the system and describe types of facts that want to be discouraged to be expressed in the displayed visualizations, these are paired with a

weight that represents the cost, within the model, of violating such a constraint. In Listing 5.2, the example given is a soft constraint that negatively weights the visualizations if a field *N* with a categorical type is mapped to the color encoding and simultaneously has a number *U*, of unique values, superior or equal to 11. This constraint is useful since the system's unique color limit is 10, this was set in an effort to safeguard the expressiveness of future possible visualizations.

```
@soft(categorical_color_cardinality)
preference(categoricalcolorcardinality,F) :-
    attribute((field,name),F,N),
    attribute((field,unique),F,U),
    attribute((encoding,field),E,N),
    attribute((encoding,channel),E,color),
    helper((encoding,scale_type),E,categorical),
    U >= 11.
```

Listing 5.2: Example of a soft constraint that was added to the system during the testing and development phase.

Similarly, but not necessarily identically, **hard constraints** distinguish what types of visual features, or combinations of these features, are prohibited from being drawn into the visualizations. Listing 5.3, shows a hard constraint that forbids a channel *C* to have two aggregate functions. This constraint was created because, sometimes, the system would apply, for example, a sum function to a previously applied count function. This behavior would create intelligible visualizations.

```
@hard(dual_aggregate)
violation(dual_aggregate) :-
    entity(encoding,_,E1),
    entity(encoding,_,E2),
    E1 != E2,
    attribute((encoding,channel),E1,C),
    attribute((encoding,aggregate),E1,_),
    attribute((encoding,channel),E2,C),
    attribute((encoding,aggregate),E2,_).
```

Listing 5.3: Example of an hard constraint that was added to the system during the testing and development phase.

In summary, when observing each of the analytical tasks individually, they possess **soft** and **hard** constraints that reflect their preferred type of mark (e.g. bar or line), relationship with the different data types, preference regarding aggregating functions (e.g. sum, average, count, etc...), preferences of certain charts to specific distributions of data (e.g discrete/discrete, continuous/continuous, discrete/continuous), usage of facets, and favorable encodings.

Relationship strive for accuracy taking advantage of scatterplots, line charts, and bar charts with size and color, having adequate encodings.

Compare seeks bar charts, side-by-side or faceted, and heatmaps with colors providing crucial visual support while usually employing aggregations.

Lastly, **Determine Distribution** does not have a true preference for charts, besides bar charts, but does have a preference for aggregating fields. Especially since the task distribution is typically connected with the count of records which commonly happens when a single field is instantiated into a visualization.

5.2.2 Achieve adequate results

Following the correct framing/scoping of the soft and hard constraints, it is vital to set these constraints' respective weights. The weight associated with a constraint can be seen as an assigned penalty if violated by a visualization. Ranking of visualization is executed by selecting the best visualizations, that respect the specifications from the input file and have the least amount of weight.

Draco uses a learning-to-rank approach, more specifically RankSVM [48], to determine how a **dataset composed of positive and negative ordered pairs of visualizations**, conveys the information and allows for the weighting of said constraints. The ranking system, a support vector machine (SVM), trains on visualization ordered pairs with a complementary set of existing soft constraints. Individual visualizations, within a pair, are turned into feature vectors when the dataset is run through Clingo, where the cost of each visualization is set by counting how many default soft constraints (constraints set by the system's authors based on visual perception research [70, 135, 134, 138]) are violated by a visualization.

After deriving the new dataset, that solemnly represents feature vectors, RankSVM trains the constraint's weight by performing linear regressions over the new dataset. The algorithm loss function, characterized by L, is used in the minimal found solution W that represents a new learned weight for a soft constraint, as shown in formula 5.3.

In the first iteration of the Draco model, Draco 1 [77], the authors expand on the utilization of Draco-Learn [126] as a tool that can learn information directly from data and how to use it for weight learning. Although utilizing this technology could have been beneficial, executing the tool to function and obtain proper results was impossible. Notwithstanding, the training using a method that sets constraints and their weight through data (data-driven) would need a large and very expressive dataset, which would be demanding and time-consuming.

$$L = \frac{1}{n} \sum_{i=1}^{k} \max(0, 1 - y_i \mathbf{w}^T (\mathbf{x}_{i1} - \mathbf{x}_{i2})) + \lambda \|\mathbf{w}\|_2$$

$$\mathbf{w}^* = \text{argmin}_{\mathbf{w}} L$$

Figure 5.3: The formula represent the loss function (L) and the weight vector ($\mathbf{w}$) that minimizes the loss function [77].

Foreseeing the necessity of determining weights for the newly defined soft constraints, created through an iterative process of identifying necessary features to adequately answer the domain questions and solve general problems with visualizations. These experiments resulted in several soft and hard constraints that try to cater to our taxonomy, taking into consideration the knowledge surrounding visualization design which is composed of available literature and the team experts' opinions.

The first step was to determine the tasks that were being included and have a set of questions that encompassed multiple possibilities and combinations of attributes from a single dataset. The initial logic was to create a dataset based on questions, for each analytical task, that focuses on tuples or triplets (sequences) of data types (e.g. (Discrete, Continuous), a pair that represents a question analyzed by a discrete field in the x-axis and a continuous field in the y-axis).

Individual tasks got multiple data sequences. Each sequence relates to a domain question relative to a specific dataset, with X possible visualizations. The following is the organization of lists of pairings:

- **Relationship :** has the pairings (Continuous, Discrete), (Continuous, Continuous), (Discrete, Continuous, Continuous), and (Continuous, Discrete, Discrete).

- **Compare :** has the same pairings as the relationship task, (Continuous, Discrete), (Continuous, Continuous), (Discrete, Continuous, Continuous), and (Continuous, Discrete, Discrete). However, the questions and subsequent visualizations are fundamentally different due to the nature of the two analytical tasks.

- **Determine Distribution :** Finally, this task is the only one capable of correctly evaluating questions with a single data field. Therefore, besides the pairings equal to **Relationship** and **Compare** it was also added for the single data fields (Discrete) and (Continuous), which can be explored visually using aggregation functions like count.

It is important to emphasize that in these instances, both good and bad visualizations are needed since a visualization pair comprises one positive and negative visualizations, to create an entry in the dataset. The dataset was created by generating visualizations in Vega-Lite [104] following the Draco specifications. Draco offers an in-built function that turns Vega-Lite visualizations into Draco format-specific files.

After having all the designated visualizations it was necessary to define a metric to which the pairs should be created. The first dataset was created based on the idea of **pairing visualizations** like such: the **highest-ranked visualization (positive)** is paired with the **second-highest visualization (negative)**, the **second-highest visualization (positive)** is paired with the **third-lowest visualization (negative)**, and so on until no visualizations remained. This approach granted relatively bad results since the new weights comprised really high and even some negative values, which should not happen.

After further studying the system, it became clear that a new approach would have to be adopted to achieve better weight results. **The new methodology was to prioritize the usage of good visualizations and pair them with less favorable visualizations**. Therefore the new methodology follows the following pair-building principle: the highest-ranked visualization (positive) is paired with the worst-ranked visualization (negative), the second-highest visualization (positive) is paired with the second-worst visualization (negative), and so on until no visualizations remain. The results showed a considerable improvement but still showed a considerable discrepancy compared with the default system weights.

It was hypothesized - although not confirmed - that these disparate results arose from a lack of training data. Due to the difficulty in extending the existing dataset much further, it would be hard, ever to obtain favorable results solemnly achievable from a dataset created by myself.

Therefore, the solution was to apply a normalizing function, which takes into consideration the default system weights and sets the lowest weight as the lower-bound and the highest weight as the upper-bound, to the existing weight values. This function was then applied to all new weight values and it helped solidify a better foundational basis.

However, it must be stated that since the weights were achieved using a blend of a machine-learning technique with a new human-defined metric, created by myself. This approach does not constitute the soundest way to attain weight values. A potential future work regarding this recommendation system would be to create a broad and expressive dataset capable of achieving good weight results without the need for extra human intervention.

5.2.3 Automation of visualizations

Creating an automatic way of producing data visualizations, by utilizing all the aforementioned work performed throughout this dissertation, came as a rational observation from the point of view of proving the possibility of conceiving visualizations based on our model-driven hypothesis. Positively connecting these different components was conducted through pipelining the dataset, data mapping, and analytical tasks available in the data repository, as seen in section 4.3, directly into the recommendation system.

In order to utilize all this information, when assembling the input file, it is necessary to parser this data and keep the essentials. The file, and how the file is written, subsequently influence the system-generated visualizations.

This pipeline program browses the data files and interprets the data present in multiple files, using a dataframe structure to access the original data easily. Afterward, it takes into consideration a description of the domain's conceptual model, translates its classes and data types onto the data fields that are present in the dataset. Matching data is kept and organized, advocating the principles of data type and data categories, to be mapped into their respective **fields**, **encodings**, and **scales** when written to the file that the system will

execute.

Following the mapping of data, it then comes the need to evaluate the analytical tasks that are associated with a specific domain question. Such tasks are executed in a set order that follows the meta-model's conventions.

The strategy for processing these individual tasks, which are divided into visualization-creating or data-processing tasks, is defined in the body of the program itself. **Data-processing tasks**, are executed mainly by the parser program, except for the analytical task aggregate. This is done through the evaluation of formulas present in the data repository. For example, a compute attribute analytical task file has information regarding the necessary formula to derive a new attribute. Whereas, available **Visualization-creating tasks**, are naturally managed by the recommendation system using the rules and constraints that were set to accommodate them. Since the recommendation system creates a visualization based on a single visualization-creating task, then each example is restricted to having a single instance of these tasks. These tasks result in visualizations illustrating the system's reading of a specific arrangement of analytical tasks and data fields.

Therefore, it is **possible to employ this automation support tool in conjunction with the recommendation system to not only create visualizations but also to query and process data**.

5.2.4 System's example demonstration

Observing a demonstration of the system, at work, will offer insight into how these visualizations come to fruition, as an extension of the work and knowledge drawn from this dissertation. Studies dedicated to analytical questions are sometimes used as a baseline in literature, by providing functional examples as comparisons. Kobsa's comparative survey [58] offers multiple questions and their respective answers to three independent domains. In our case, it was chosen the dataset composed of technical data of cars [10] and the picked domain question was: "Which manufacturer produced the most cars in 1980?".

Analysis of the domain question and dataset, suggests using an ***Aggregate*** task - which will count every instance of the attribute "brand" in the year 1980 (The meta-model allows for every analytical task to have a filtering formula associated to it) - and a ***Determine Distribution*** task - where the distribution of values from the count function will be tied to each brand, in a single visualization - would be an adequate answer.

Considering that the dataset, conceptual model, and designated tasks are kept in the data repository and, more specifically, the latter are translated into JSON files that store significant information for the recommendation system. Parsing and processing this data, an analytical task sample can be seen in Listing 5.4, will result in an intermediate file, with transformed structured-data, that when executed by the system renders the visualization seen in Figure 5.4.

```
{
```

```
questionID: 1411,
taskID: 14111,
task_selection: [
        {
         filter: "featured_dataset[featured_dataset['year']==80]",
          task: "Derive",
                subtask: [{task: "Aggregate", type: "count", attribute: "brand",
                    ↪ groupby: "brand"}],
                domain_specification: "Which manufacturer produced the most cars
                    ↪ in 1980?",
                analytical_specification: "Count the number of cars produced,
                    ↪ per manufacturer, in 1980.",
                expected_output: "List with the number of cars produced, per
                    ↪ manufacturer, in 1980."
              }
      ]
}
```

Listing 5.4: Example of a JSON file characterizing a single analytical task (**Aggregate**).

Examining the visualization in Figure 5.4, allows us to determine that the answer to the domain question is Datsun and Volkswagen, both tied with four occurrences, which the study confirms as the correct answer. This small example shows a singular result of an active effort that supports our approach and partially confirms our initial hypothesis. Although the visualization provided by the system is adequate, it could improve slightly with the following enhancements: **(1)** - Having the marks ordered in an ascending/descending manner; **(2)** - Displaying the bars horizontally, instead of vertically, for easier mark and value readability.

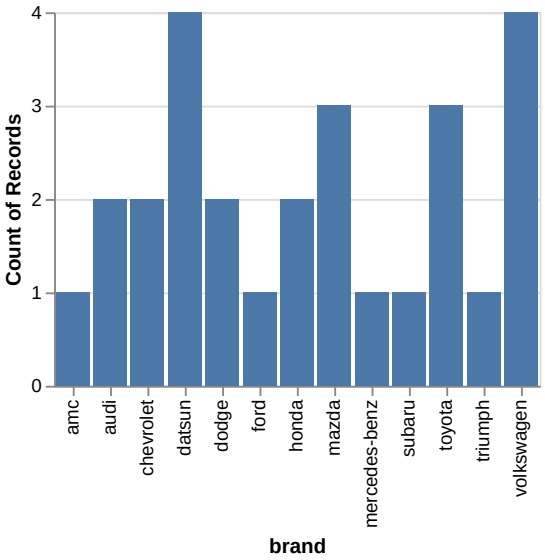

Figure 5.4: Best-ranked visualization to answer the domain question **"Which manufacturer produced the most cars in 1980?"**, according to the recommendation system.

5.3 Overview

Summarizing this chapter, the visualization recommendation system chosen, as a possible tool to visually demonstrate our approach, was Draco. The system needed to fit within our designated guidelines, present in Section 5.1, and Draco managed to match our expectations. Showing itself as a valuable instrument during the development of an environment capable of supporting our model-driven methodology.

A system expansion was deemed crucial to add internal and external features to the system, such as the implementation of the proposed taxonomy, visualization training, and automation, which were introduced as augmentations to the existing recommendation system.

Although Draco was capable of expressing a sufficient number of visualizations and visual encodings, it was not enough to fulfill all of the possibilities present in the meta-model. Thus, it is reasonable to state that the system is somewhat restricted in terms of visualization and aggregation options (by default Draco offers a set of available data aggregating functions). Furthermore, the system was also made to handle easier visualizations with reduced data fields. Another limitation is tied to the fact that the system can only execute a single visualization-creating task per input file.

Such limitations can be diminished with further development work into Draco or by adopting another recommendation system, which might yield different data processing and visualization characteristics.

6

Testing & Evaluation

This chapter will address testing and the respective evaluation surrounding the components developed during the dissertation. Acknowledging the scope and focus of this dissertation, the consensus approach was to conduct various case studies, to achieve the most appropriate testing. Thoroughly analyzing each instance while connecting the individual results to the positive and negative aspects of this dissertation's methodology will be a key point in this section.

In an ideal data visualization scenario, the first objective would be to pinpoint the correct translation of domain questions into abstract analytical questions. This abstract question would need to be further derived/translated into sets of analytical tasks, capable of answering the initial domain question. However, in light of easing the development, translations were made to be directly executed from a domain question to a set of analytical tasks, without the need to attain the abstract analytical questions. Thus, this transformation could be conveyed by either a data specialist or a LLM.

Given these considerations, this dissertation's testing and evaluation will be performed in two distinctive aspects:

- (1) **Translation** - Indicates how the meta-model and the taxonomy work in symbiosis, to lay a foundation for case studies of different domains that will have the translation of domain questions directly into analytical tasks, done through different LLMs. Their answers will be compared with the expected answers derived by human data experts, and their effectiveness in answering the domain question.

- (2) **Visualization** - It will take advantage of the whole theoretical and practical developments, by promoting the usage of the previously discussed components throughout this thesis, (i.e. Meta-model, Taxonomy, LLM Translation, Data Repository, Visualization System, and Automation Module), to achieve resulting visualization and discuss how these visualizations perform in answering the domain questions.

Ultimately, this chapter is intended to provide a clear insight into **What** the methodology is able to do, **How** it is capable of dealing with real examples, and finally **Why** this approach makes sense.

6.1 Task Translation

The tests regarding translation will take into consideration five unique domains that are divided into multiple instances of questions. The goal will be to examine, evaluate, and compare the answer with the expert's expected answers counterparts.

Tests were comprised of two batteries: A first set of tests was performed to understand if it was possible to utilize LLMs as a supportive tool for this important process in data analysis and data visualization; The latter was created with a deeper level of understanding regarding these technologies.

Testing was agnostic in regard to decoding strategies/techniques, these were not taken into consideration. Furthermore, it was done assuming that the default parameters used by each model were sufficient, even if the baselines between them were not the same due to inherent differences between the models.

6.1.1 Phase 1 Translation Testing & Evaluation

In the first testing sample, executed solemnly using the model GPT-3.5 [88], each domain had a variable number of questions with an assortment of different difficulties. The questions were evaluated following a single metric: whether the answers provided by the model were comparable to those provided by the experts.

Table 6.1 displays a few examples of the LLM evaluation and translation present in the **Phase 1** data pool of domain questions, with their respective evaluation. The model needs to absorb a significant part of the context regarding the domain, taxonomy, dataset, and domain question to attain the most fitting answers.

Therefore, before giving the model any questions it is paramount to provide it with all the available context. The initial prompt bears the problem's context and objectives, where a clear explanation of the domain and what we want to achieve is given. After the model is capable of recognizing the new context, it is then required to assimilate our proposed taxonomy, and a sample of the domain's dataset as the remaining context to take into consideration.

From that point onward, the subsequent prompts are the domain questions together with the request for the model to answer them using the aforementioned analytical tasks. This was our rule of thumb for testing in this phase. **A sample from a conversation log can be found in annex I.**

Our resulting ground truth document organized these individual experiments into four categories:

- **Domain Question**: Domain question inquired, found in the prompt.
- **Model's Answer**: The answer provided by the model, proposes the necessary tasks and explains the step-by-step rationale.
- **Expected Answer**: The answer the experts expected to be correct.

- **Evaluation**: Displays the accuracy of the Model's Answer when compared to the Expected Answer. The metric was labeled as a value ranging from 1 to 5, where one represented an answer with distinct tasks and five represented a perfect match. All values below 3 were considered unsuccessful answers, whereas 3 and above were considered to be successful answers, even if they were sub-optimal.

Domain Question	Model Answer	Expert Answer	Evaluation
"Are there variations in the average discount rates across different product categories?"	1. **Derive**: *Aggregate*: Calculate the average discount rate for each product category.	1. **Derive**: *Aggregate*: Average of the discount rate grouped by each category. 2. **Compare**: Compare all categories average discount values.	**2**
"How many cities are represented from each country?"	1. **Derive**: *Group*: Group the cities in your dataset by their respective countries. 2. **Derive**: *Aggregate*: Aggregate data within each country group to count the number of cities in each country. 3. **Determine Distribution**: Determine the distribution of the number of cities per country.	1. **Derive**: *Aggregate*: Count the number of cities per country. 2. **Determine Distribution**: Check the distribution of cities per country.	**4**
"Which type of genotypes tend to be associated with a G allele?"	1. **Identify**: *Relate*: Identify the relationship between the genotypes and the presence of a G allele. It will help determine which genotypes are associated with a G allele. 2. **Determine Distribution**: Allows analysis of the distribution of genotypes and the presence of the G allele. You can identify how often each genotype is associated.	1. **Identify**: *Relate*: Filter by G allele and relate the connection between the genotypes present.	**3**

Table 6.1: Examples of different ground truth entries with domain questions, answers, and respective evaluation.

Observing the Table 6.1, shows examples of three of the five available domains, with differences in their evaluation result. Analyzing the **first example**, its evaluation was a **2** since it lacked the ***Compare*** analytical task, that would allow the user to visually judge correctly the variations between the averages and obtain an appropriate answer. In the **second example**, the evaluation score received was a **4**, although the model provided two correct answers it suggested an extra non-visualization-creating analytical task compared to the expected answer, and as such received an evaluation penalty. Lastly, the **third example** received an evaluation of **3**, even though it features the expected visualization task it also indicated an additional analytical task deemed unnecessary.

After documenting all the results and analyzing the various domains, as it can be seen in Figure 6.1, these conclusions ensued:

When considering the results, in the **temporal domain**, the model reached a successful ratio of **57.1%** (received an evaluation of 3 or above), the best-performing domain in **phase 1**. The prompts and respective domain questions shared a higher level of similarity

between them, because of the dataset restrictions, and this factor might have influenced the results.

The group of questions about the **business domain**, a generic and low-complexity domain, obtained slightly positive results. However, only **54.5%** of the answer achieved a **satisfactory evaluation**.

For the **socioeconomic** dataset analysis, the system returned overall **negative results**. Only **45.5%** of the answers were determined to be valid, equivalently to the temporal domain and business domain, the model might be achieving better performances because of the more simplistic nature of the domain.

Switching to a **spatial domain**, the answers were mostly negative. Most domain questions had problematic answers because the model could not understand the need to derive attributes or apply some extra abstraction layers. Achieving only a **success rate** of just **38,5%**.

Looking at even more complex domains, in this instance the **human genome domain**, the model became much more fragile and generated multiple answers that could not be considered acceptable. It showed difficulties in understanding and adapting to the more abstract concepts, acquiring only **28.6% of successful answers**.

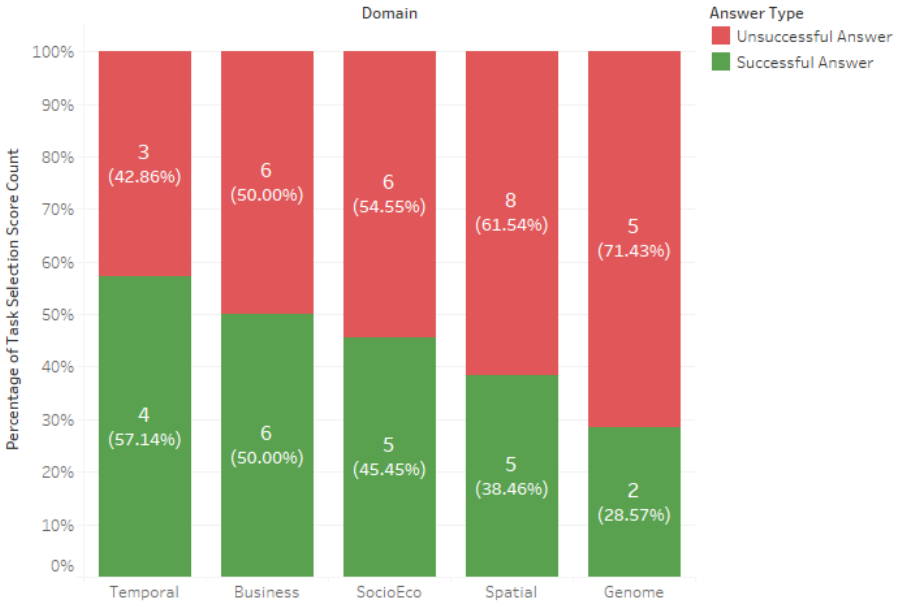

Figure 6.1: Number of occurrences of **successful** and **unsuccessful** answer in **Phase 1**, ranked from best to worst.

The results present in Figure 6.2, show **an evaluation score above 3 in two of the five domains**, which supports the values seen in Figure 6.1. Although these results are, on average, below the positive line, it is important to acknowledge the existing limitations surrounding these results (i.e. Low number of questions, zero-shot learning, and lack of structure).

During this testing phase, there was no clear intention of classifying these results beyond observing their potential translating capabilities. They serve as a baseline to

understand how **LLMs** might be able to fulfill some of the needs within this dissertation's strategy.

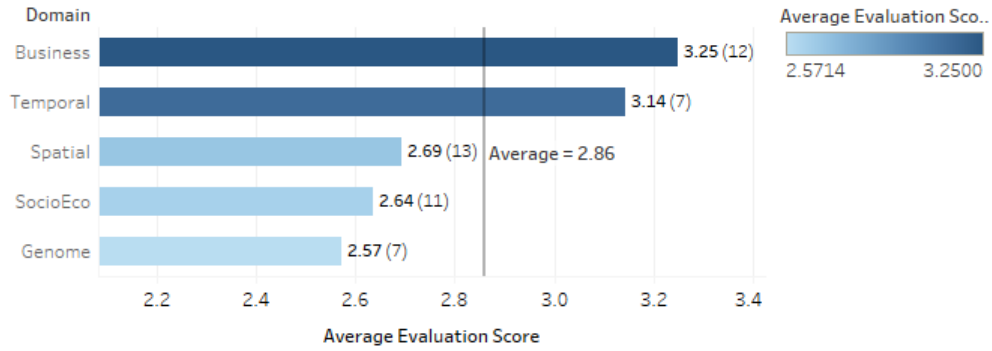

Figure 6.2: Distribution of the average **Evaluation** score in **Phase 1**, for each tested domain, order from best to worst-performing, and the respective number of questions answered.

However, we believed a more refined procedure in conjunction with a richer number of questions, could yield better results and serve as acceptable proof to take measures about future work and analysis with these technologies.

This will be the guiding hypothesis in **Phase 2** of testing, in the next Section 6.1.2.

6.1.2 Phase 2 Translation Testing & Evaluation

After the results obtained from the earlier translation tests, it was understood that large language models had a sufficiently good basis to work as a translation support tool to achieve analytical tasks. Although this was the initial premise, the lack of models that were tested and evaluated as well as the non-existing test structure contributed to a set of tests that could be considered incomplete.

One significant detail that guided us during this **second phase of testing** was that **LLM** research shows that these models are known for being **few-shot learners** [122, 20]. This means that when given multiple examples, **LLMs** tend to produce better results (somewhere between 10 and 100 examples, because of the model's context window) [20]. There are four types of "learning" methodologies when dealing with language models:

- **Zero-shot Learning**: The model is given no previous examples and is expected to know how to behave when confronted with the user prompt. It is typically the worst-performing strategy with context-based knowledge.
- **One-shot Learning**: A specialization of "zero-shot", in the sense that the model receives one single demonstration to understand the needed context. This approach mimics how information is communicated to humans in an everyday setting.
- **Few-shot Learning**: An approach that offers the model some ongoing examples, finished, or completion examples. The main advantage is the reduction of non-task-specific data and the creation of a more broad model. Unfortunately, the downside

is that the results will typically reach a plateau where they can not perform any better.

- **Fine-tuning**: A very specific type of training, since it involves giving labeled data that focuses on a specific context or set of contexts, to change the training weights of the model.

In the **first phase**, our direction started with a zero-shot learning approach, giving us a baseline to understand how the model behaves when in contact with this given context. Our primary approach in **Phase 2** will serve as a new standard, using a few-shot approach, which ultimately will be compared to the results obtained in **Phase 1**.

Addressing a new methodology would come as a necessity to demonstrate the true effectiveness of different models adapting adequately to domain question translation. To create a more complete testing environment a new methodology was framed: each of the five existing domains has a clear dichotomy of **Direct domain questions**, where the intended task is easy to detect within the question itself, and **Interpretative domain questions** which tend to have a domain-specific language that the model must contextually understand to select a suitable set of tasks.

Due to the need to provide a more in-depth evaluation, compared to the previously executed tests, the old evaluation was revised and transformed into three separate categories:

- **Domain Knowledge** - Judges the model's ability to correctly read the domain and how in-depth the response was. This category allows a range of values from 1 to 5, where the lowest score represents a bad interpretation of the contextualized domain while the highest is an excellent interpretation.

- **Question Understanding** - Classifies how coherently the model appeared to be in its response to the domain question. Utilizes an interval of values from 1, which discloses a bad comprehension of the question by the model, to 5, which shows a great grasp of the question's context.

- **Task Selection** - Classifies the quality of the answer given by the model, similar to the previous labels it ranges from 1 to 5, where 1 and 2 determine an unsuccessful answer, and any evaluation 3 or above is a successful answer (potentially, one of many existing answers).

Domain Knowledge and **Question Understanding** are two evaluation methods that are not entirely concrete, as they inherently rely on a degree of personal interpretation and judgment. Consequently, their results come as a direct influence from personal experience and understanding of how these technologies absorb information and how they contextualize said information.

In this second testing phase, a primary point of focus - besides the endorsement of new evaluation categories - was the selection of multiple large language models to

broaden the approach's perspective and achieve more homogeneous results. The models GPT-3.5 [88, 20], BingAI's GPT-4 [74, 89], Mixtral 8x7B [50], GPT-4 [88, 89], and Gemini 1.0 [41, 115] were chosen as means to have a wider range of testing options, with a multitude of answers to scrutinize.

Contrasting with Phase 1, the structure of the second phase of tests, more specifically the individual questions, was organized following a **Direct** and **Interpretative** set of questions. The **Direct** questions always came in a set of 5 questions and **Interpretative** ranged from 5 to 10 questions, depending on the domain.

6.1.3 Direct & Interpretative Question Evaluation

Direct and Interpretative question evaluation of Task Selection, referenced in Figure 6.3, presented good results in the overall number of successful answers. The **direct questions** had a **success rate** of **66.4%**, which can be considered a very positive result. It is a starting leap towards achieving better results and possibly more adequate translations, at least, for direct/simpler questions. **Interpretative questions** had a slightly worse percentage ratio when compared to the direct ones. Nevertheless, these were still considered good. Reasoning over the achieved results, it was very promising to hit a **success rate** of almost **60%**.

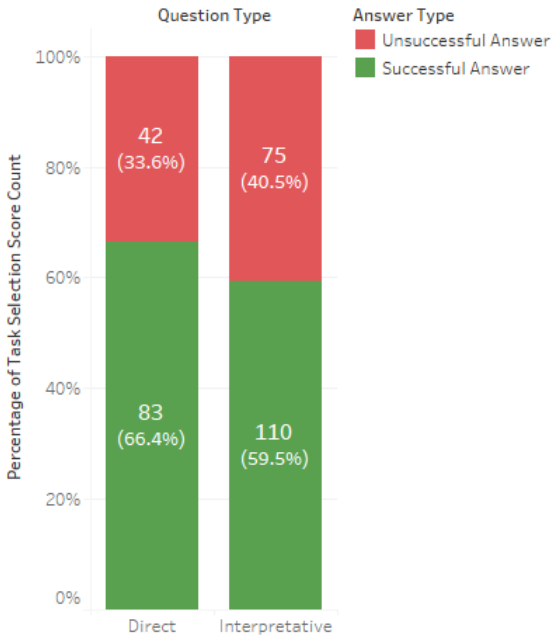

Figure 6.3: Number of occurrences of **successful** and **unsuccessful** answers in **Direct** and **Interpretative** questions in **Phase 2**.

When analyzing the results with our methodology in mind, it becomes evident that direct questions achieved slightly better results, when compared to interpretative questions because the inherent differences in difficulty, between question types, hinder the overall results of the models. The lower-than-average difficulty nature of **direct questions** is

usually associated with questions such as: "**What is the total profit value for each sub-category of products?**. Whereas, the more difficult **interpretative questions** came to fruition as questions that distinguish themselves from the **direct** ones by possibly having additional layers of abstraction. **Interpretative questions** typically were created with the intent of needing some type of extra knowledge or implicit information in the question itself, for example: "**Identify potential hot spots for existing variants.**".

Projecting the other two evaluation categories, **Domain Knowledge & Question Understanding**, that represent the quality of answers in terms of domain and question as depicted by the surveyed models. Figure 6.4, shows a scatterplot that relates the domain knowledge and question understanding evaluation, domains, and types of questions.

This visualization determinedly exhibits that **LLMs** do a good job at contextualizing and reasoning about domains and their existing questions. In spite of the fact that such a metric is hard to quantify, in almost all instances, the extra information within the model's output allows for an educated evaluation of how good, or bad, the model's reasoning was.

Regarding Figure 6.4, the trend set by Figure 6.3 continues to be true because it is possible to observe that **direct** questions continue to have higher evaluations than the **interpretative** counterpart, in most instances.

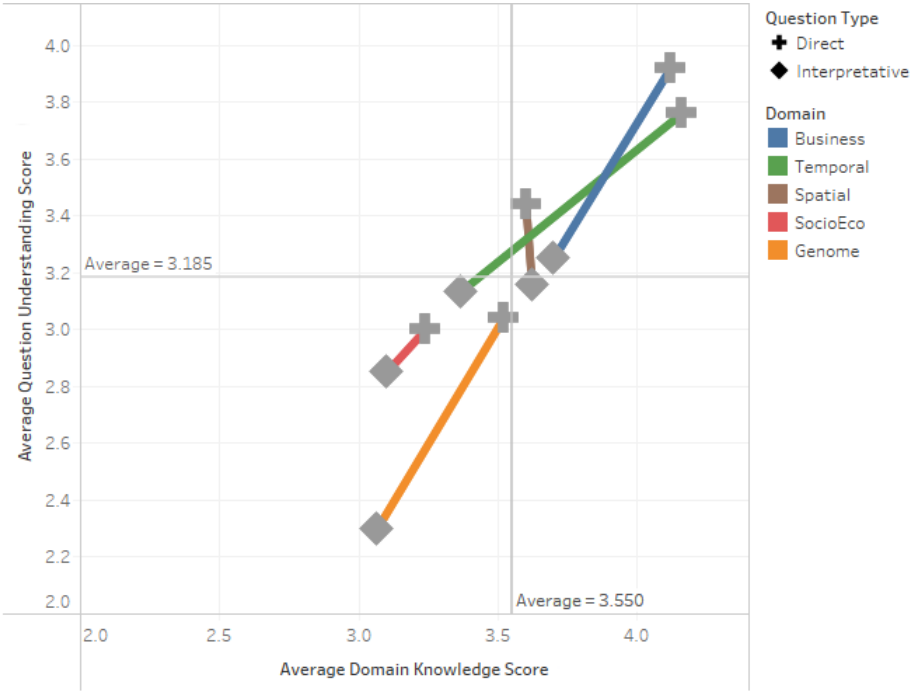

Figure 6.4: Visualization that relates the **domain knowledge** and **question understanding** scores with **domains** and **question type**, exclusively in **Phase 2**.

Domains and question types, in this visualization, reveal that the **Genome** and **Temporal** domains were hard to reason. The **domain knowledge** and **question understanding** averages of 3.18 and 3.55, respectively, are encouraging since they show that, on average, models are correctly addressing the domains and the corresponding domain questions.

The averages come as a logical development since harder questions would have, on average, worst-scoring answers from the models while simpler questions have, on average, a best general performance. This is mirrored in our results, which suggest that, holistically, our strategy and methodology had a sound foundation.

6.1.4 LLM & Domain Evaluation

The five tested **LLMs** and **Domains** within this testing section were a target of analysis since they played an important role in the construction of our ground truth.

Domains serve as a variable that influences the model's results due to their differing natures and difficulties. Examining the data, exclusively related to **Phase 2**, sets a landscape that allows for some domain scrutiny. In Figure 6.5, all domains have the total number of successful and unsuccessful answers given by models in **Phase 2**. These results show that models tend to prefer common domains, where they achieve the best results, such as business and spatial. However, it becomes apparent that more intricate and complex domains, e.g. the human genome, make models struggle.

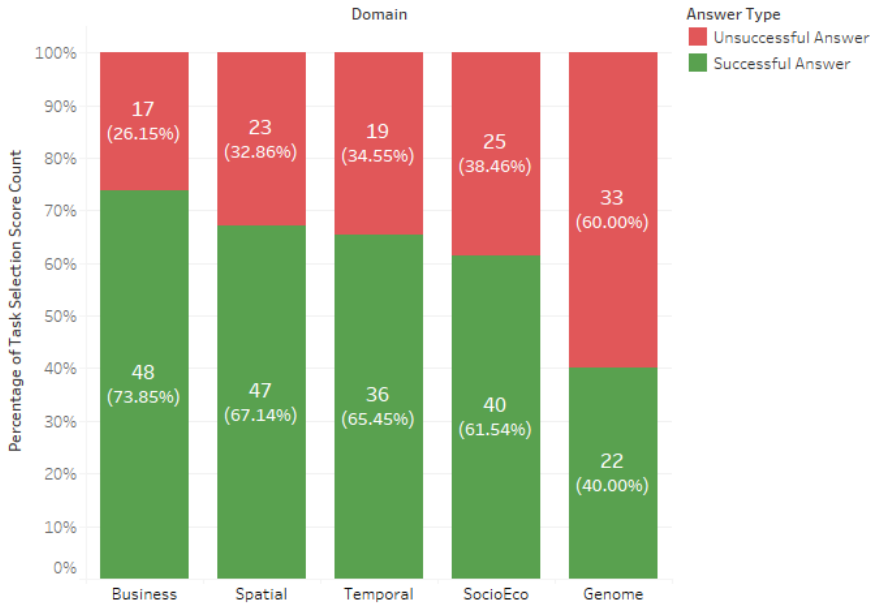

Figure 6.5: Distribution of **Phase 2 successful** and **unsuccessful answers**, for each domain, ordered by their successful answer percentage.

Looking specifically at Figure 6.5, models were able to answer convincingly to almost all domains. **Business** and **Spatial** are the domains with the highest amount of successful answers, 48 **(73.85%)** and 47 **(67.14%)**, respectively. Closely followed by the **Temporal** domain, which had a smaller number of domain questions, with 36 **(65.45%)** successes, and the **Socioeconomic** domain, with 40 **(61.54%)** successful answers in all domain instances. On the flip side of the spectrum, the **Genome** domain, the most complex domain, was the only one with an overall negative record of 33 **(60%)** unsuccessful answers, and only 22 **(40%)** successful answers.

Regarding the evaluation of the domains' **domain knowledge** and **question understanding**, that was previously seen in Figure 6.4, it shows that domains typically have a big difference when they are evaluated and reasoned through the usage of direct questions or interpretative questions. This reasoning coherence from the models makes sense since it cements our hypothesis that models do perform better when answering easier domain questions, in easier domains. In that same visualization, it is clear that some domains **Temporal** and **Genome** have a pretty sizable gap in terms of difficulty with regard to **direct** and **interpretative** questions. Consequently, this means that the model typically will perform worse with very specific domains that are also capable of performing various degrees of questions. For example, the two domains **SocioEco** and **Spatial** have shorter relative differences because their datasets did not allow for highly in-depth questions.

The ground truth experiments with **LLMs**, executed in **Phase 2**, create a generalized model review that utilizes the three evaluation categories as criteria. One of the lines of work for this evaluation phase was also understanding the viability of including a fine-tuning learning methodology for domain question translation, as possible future work. Therefore, we are interested in scrutinizing the results and identifying the best-performing model, that could be considered the most suitable for future work reference.

Figure 6.6, in general, demonstrates that the **models seem to be sufficiently able, on average, to translate domain questions into analytical tasks**.

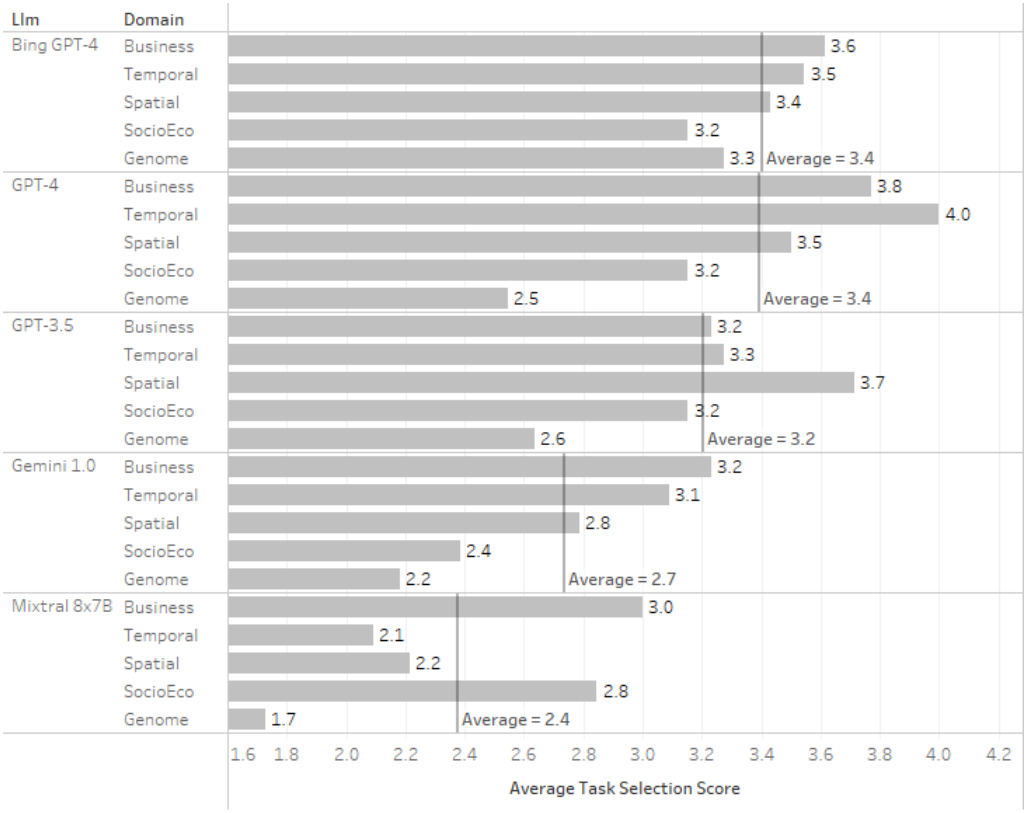

Figure 6.6: Distribution of models' average **task selection** score, in **Phase 2**, for individual domains, from the best to the worst-performing model.

Individually examining each model's average task selection performance, for each domain, gives a slightly different impression. Judging by averages alone both **Bing GPT-4** and **GPT-4** are tied as the best-performing models, and **Mixtral 8x7B** and **Gemini** were the worst-perfoming models, with a significant distance to the remaining three. Dissecting these numbers even further the values, it is possible to observe that **GPT-4** has a bigger range of evaluation results, both positive and negative. In contrast, the **Bing GPT-4** model was more precise, with a smaller fluctuation of values. The model **GPT-3.5**, the only one used in **Phase 1**, improved all averages in comparison. This is a great indicator that our **Phase 2** strategy, with an improved structure and the implementation of few-shot learning, is capable of yielding positive results.

Another layer of evaluation is the assessment of the **domain knowledge** and **question understanding** of each model, which creates a new dimension to which the models can be compared. Figure 6.7 tackles these remaining categories of evaluation.

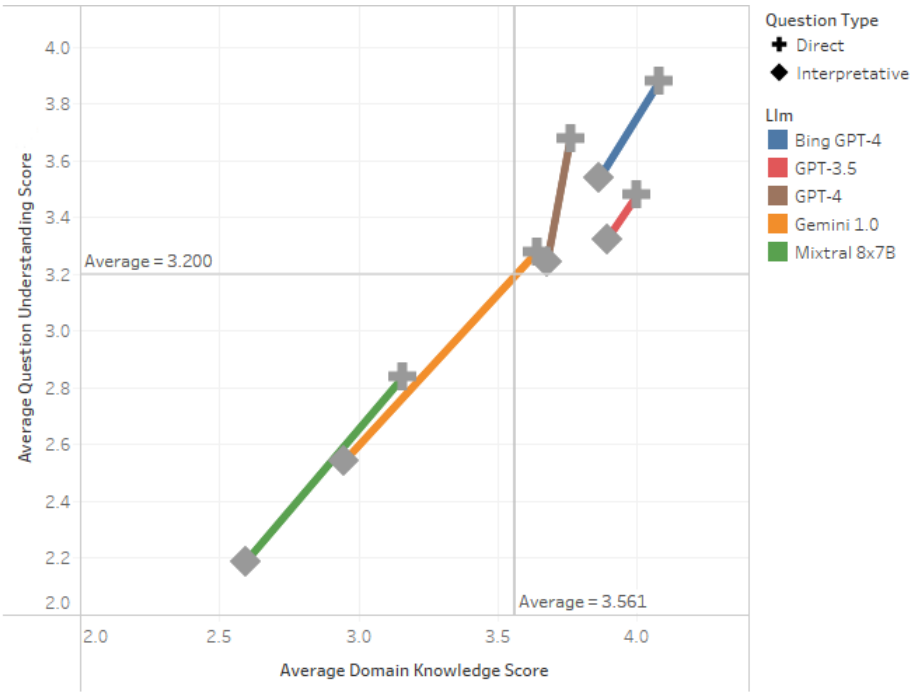

Figure 6.7: Visualization that relates the **domain knowledge** and **question understanding** scores with the **LLMs** and **question type**, in **Phase 2**.

Exploring this visualization allows the determining of **Bing GPT-4** as the best-performing model and **Mixtral 8x7B** as the worst-performing. The visualization also conveys the idea that the best-performing models fluctuate slightly when answering **direct** or **interpretative** questions. Conversely, the worst-performing models create a big differential in answering quality when confronting different question types.

As a whole, the LLMs proved to be capable of understanding both the questions and the domains at hand, when analyzing individual domain questions.

6.1.5 Translation Conclusions

As a summary, translation testing proved to be an important knowledge resource that might result in future ramifications. In this section of testing, which was further divided into two phases, was performed to understand both the potential, done throughout **Phase 1**, and baseline, accomplished during **Phase 2**, of **LLMs** translation.

Phase 1, although smaller in size and perspective, was done with the intent of knowing the extent to which LLMs could work in such an environment. Due to its limited nature, this phase drew few conclusions that became important later on as a driving force for **Phase 2**.

The single model tested in this phase, **GPT-3.5, performed sufficiently in two of the existing five domains and attained a negative average evaluation score**. These results were not great but set some expectations of how these models could execute with a better methodology and strategy. Based on these conclusions, **Phase 2** was hypothesized by analyzing this technology with a studied and robust approach.

Phase 2 initiated with more evaluation categories, with two distinct types of questions, that were later determined to impact the final models' translations. **Models were tested in five domains with a positive outcome, since four in five domains submitted a total of 50%, or more, of successful answers**. In this phase, it was identified that the models' results directly correlate with the domains and type of questions (e.g. Direct/Interpretative) complexity. **Bing GPT-4** was the best performing model in all the possible categories, while **Mixtral 7x8B** proved to be insufficient for domain question translation. The **Genome Domain** showed a major limitation attached to these models' answers, the difficulty in understanding more sophisticated domains, especially if dealing with questions that require a higher level of understanding.

Overall, our strides toward understanding the viability of translation, created a baseline that might serve as a supportive argument that future work, developed upon this specific task, might be compelling and worthwhile. Our findings, especially in **Phase 2**, provide a standard that, on average, can theoretically translate domain questions into analytical tasks adequately, even if sub-optimally.

6.2 Visualization Generation

Visualization generation testing will evaluate examples from three domains, that were tested previously. The usage of different thorough examples will show how adequate the visualizations created by the recommendation system are and how the whole approach works with examples by using all developed work throughout this dissertation.

Therefore, the proposal for this section of testing is the **evaluation of individual visualizations** from **multiple domain instances** and an **exhaustive example** that will start in the beginning (conceptual model) and reach the end (resulting visualizations). Allowing for the correct assessment of the system's strengths and weaknesses.

Evaluation of individual domain examples will be limited to a visualization per instance. While the step-by-step approach, relative to the start-to-finish example will concern a broader spectrum to demonstrate how all developed components complement each other and result in a visualization.

Within these tests, our objectives are to strive to understand how the recommendation system can adapt to varying domains and the necessary specifications of each domain question. Such questions will be classified into two/three instances that create differing visualization results, where the top-ranked visualization will be detailed. Visualizations can be found, their original size, in the repository found in the annex II.

6.2.1 Domain Visualization Testing

The evaluation of resulting visualizations to specific domain questions will be addressed in three domains (i.e. Genome, Business, and Spatial). Each of these domains will present a domain question associated with it and arrange, at least, two different instances to cast diverse visualizations, under changing circumstances. In this section, the examples will be accompanied by a table that will provide insight regarding each domain, the domain question, the abstract question, the analytical task decomposition, and the instances. As mentioned previously, a single visualization will be reviewed for each domain instance, owing to the fact that these visualizations are capable of showcasing some of the system's strong points and liabilities.

6.2.1.1 Genome Domain

The genome domain focuses on the human genome, particularly, how variants, chromosomes, and genes relate. The genome dataset represents positions, in the DNA where different variations might exist, genes, and chromosomes related to each position. Variants are characterized by their relative chromosome, their DNA sequence position, reference allele, alternative alleles, that can be observed in that same position, genes affected by a variant, and the individual's genotype. Within the dataset, these concepts are defined respectively by the following data fields: **CHROM**, **POS**, **REF**, **ALT**, **GENE**, and **GENOTYPE**.

Figure 6.8 characterizes this domain's domain question, abstract question, required analytical tasks, and testing instances.

Within this domain, our objective is to find an adequate visualization to the question **"What is the distribution of variants along the genomes in the sample ?"**, which was posed by a genome expert. Studying this question further makes it possible to understand that the user wants to know the positional distribution of the different variants. This analysis, in conjunction with examining the conceptual model of the genome domain, in our meta-model, enables one to make decisions relative to which analytical tasks can answer this domain question. Since the user explicitly asked for the definition of left and right alleles, these needed to be derived from existing dataset information.

Genome Domain	
Domain Question	What is the distribution of variants along the genomes in the sample ?
Abstract Question	The user wants to know the positional distribution of the different variants.
Analytical Task Decomposition	**Compute Attribute** (Left Allele Index); **Compute Attribute** (Right Allele Index); **Compute Attribute** (L. Allele); **Compute Attribute** (R.Allele); **Determine Distribution**
First Instance - The data fields given were POS, GENE, and their respective data types. POS is declared in the x-axis, as a linear field, and GENE is associated with the y-axis, as a categorical field.	
Second Instance - Defined data fields were POS, declared as a continuous variable in the x axis, GENE and CHROM both declared as discrete variable, without an assigned axis/encoding.	
Third Instance - Defined data fields were POS, declared as a continuous variable in the x axis, GENE, Left Allele and Right Allele declared as discrete variable, without an assigned axis/encoding.	

Figure 6.8: Summary table relative to the **genome domain**, where the domain question, abstract question, analytical tasks, and domain instances are present.

Therefore, the necessary analytical tasks are: **Compute Attribute** [Derive] and **Determine Distribution**, but due to the nature of how left and right alleles are determined and calculated, the task **Compute Attribute** must be executed four times. Once for the positional index of each allele, and then the proper allele, which can be found based on its index. Since these attributes are created by specified formulas, our parser had to be able to understand them, in order to execute this analytical task, within the system.

Evaluating the **first instance** of the human genome domain, where the instantiated data fields were POS and GENE. POS was declared as a linear field on the x-axis and GENE was set as a categorical field on the y-axis. This resulted in the following visualizations:

- **(1)** - Classified as the system's best-ranked visualization, for this instance. It describes a bubble chart with POS as a continuous variable, on the x-axis, and GENE as a discrete variable on the y-axis. The visualization is positive, although it suffers from some display cluttering.

- **(2)** - Possibly the best visualization in this instance. Describes a tick chart with POS as a linear x-axis and GENE as a categorical y-axis. The visualization is adequate, it is possibly the best visualization in this specific instance. It is preferred compared to (1) since it does not suffer from the same amount of cluttering, which facilitates extracting information about each mark's position.

- **(3)** - Draws a tick chart, very similar to (2), but it bins the x-axis, which holds the POS data field. Besides the binning, the visualization also features an aggregate function to count the number of positional (POS) occurrences within each bin, this aggregate is associated with a color encoding. It is probably less expressive than (1) and (2) because pinpointing, with some degree of precision, the position of a gene is unfeasible.

Figure 6.9 displays **Visualization (2)** - The numeration does not imply an intrinsic order - which shows the best visualization present in this instance. The visualization cannot answer the question completely. Still, it provides positive insight regarding the position which is a good starting point to answer the overall question. However, the limitation imposed in this instance, of only having two data fields, hinders the capacity to extract the most information achievable.

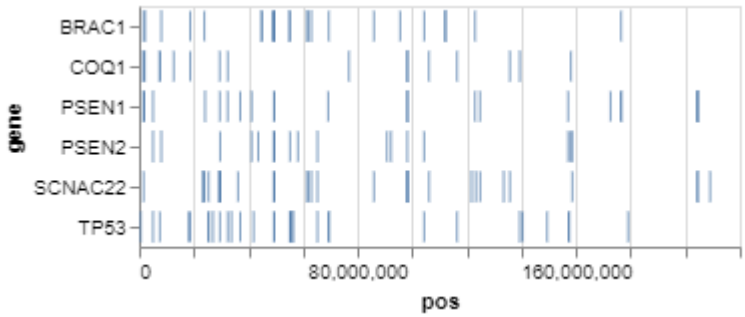

Figure 6.9: **Visualization (2)**, appears in the first instance of testing of the Genome domain.

In the **second instance**, initiation of information is close to the **first instance** with POS as a linear x-axis, but GENE and CHROM declared as discrete variables without designated axis/encodings (i.e. color, size, mark). The system generated the three best-ranked visualizations based on this description:

- **(4)** - Describes a bubble chart with POS in the x-axis, CHROM in the y-axis, and GENE was selected to be mapped to the color encoding. The bigger y-axis, with a better distribution of marks, helps with scattering and balancing the visualization space. This visualization is **better than the ones seen in the first instance** because it has more information with the extra field and is easier to identify.

- **(5)** - Applies a line chart to this analytical task. The visualization maintains the same principles seen in (4), with POS on the x-axis with a linear scale, CHROM on the y-axis as a categorical variable and GENE, once again, mapped to the color encoding. The overlapping of lines is prevalent throughout the whole display and incorrect choice of a visual mark renders the visualization inadequate and practically impossible to be useful.

- **(6)** - Renders a visualization characterized by a tick chart that maintains the same arrangement as (4) and (5). The visualization, similarly to (4), allows for a good identification of the individual positions for each chromosome and gene. Its design choices grant a **quality visualization that is able to answer the question at hand**. However, it could still improve if it emphasized its marks.

Figure 6.10, **Visualization (6)**, is perhaps the best visualization available seen until now. It has a distinctive use of the data fields and the higher amount of scattering helps

with the identification of the position distribution of variants in the available genes, further distributed by the chromosomes.

It is a direct improvement to the visualization seen in figure 6.9. Perchance it could still improve since it does not feature the left and right allele, that was asked to be included by the genome expert. The choice to not include it here was deliberate because they will be added in the **third instance**.

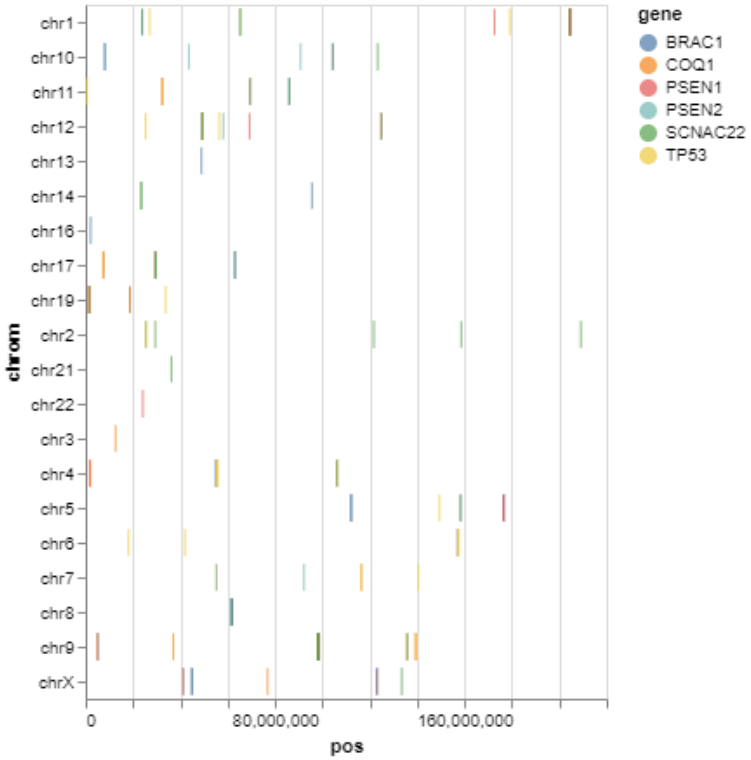

Figure 6.10: **Visualization (6)**, present in the second instance, of the Genome Domain.

The last instance in this domain, **third instance**, is characterized by having two new derived attributes. The definition of the visualizations is done through the selection of four data fields: POS, as a continuous variable in the x-axis, and GENE, Left Allele, and Right Allele all declared as discrete variables, not mapped to an axis nor an encoding. Unlike previous instances, this one only features two visualizations since the third one was a duplicate of **(8)**.

- **(7)** - Draws a bubble chart with POS and Right Allele in the x-axis and y-axis respectively. Left Allele is elected by the system to be mapped to the color encoding and GENE is ultimately mapped to the mark encoding.

- **(8)** - Structurally similar to (7), but there is an exchange between the Left and Right Allele, which are the y-axis and the color encoding respectively. GENE continues to be mapped to the mark encoding.

Visualization (7) - present in figure 6.11 - was considered to be discussed since it exposes some problems with the visualization generation by the system. This visualization uses the newly computed attributes (Right Allele and Left Allele) that were characterized but not used in previous instances, as a direct consequence of the system not being well-equipped to handle more than three or four data fields simultaneously. Especially if there are a considerable number of unique values for discrete fields.

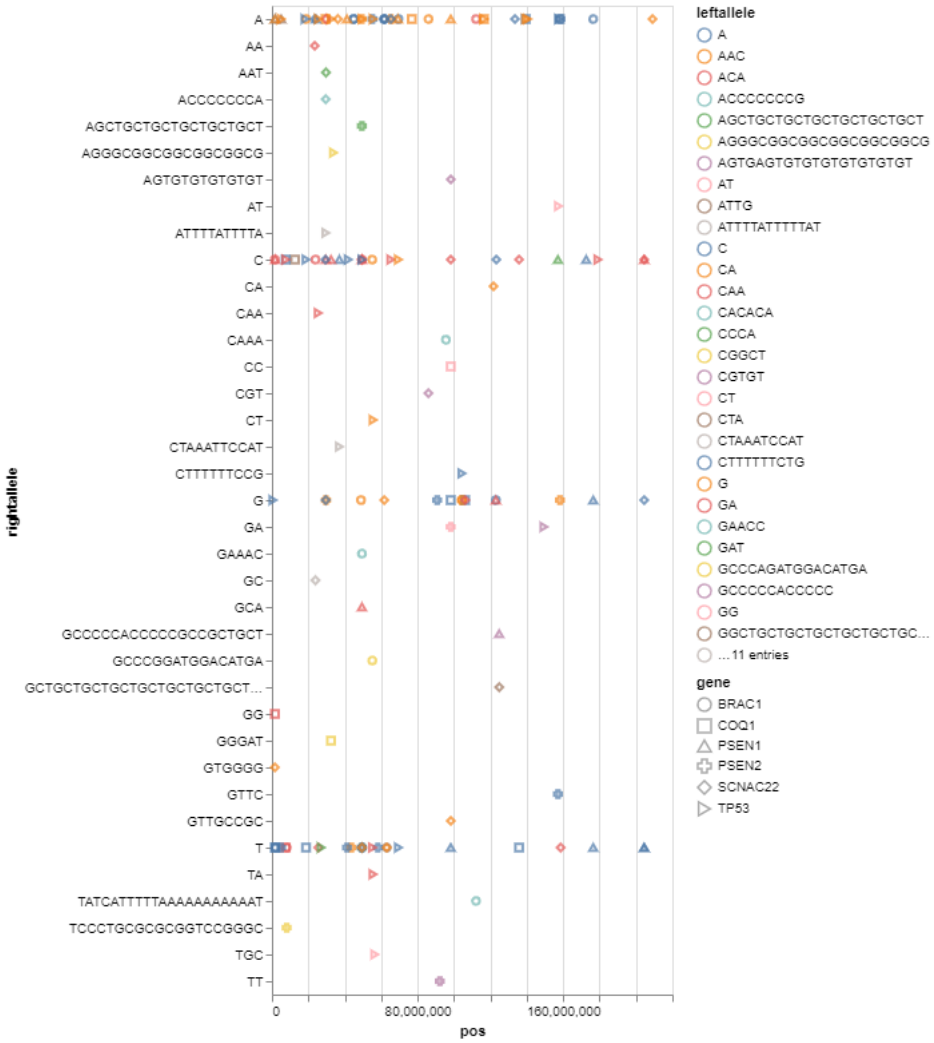

Figure 6.11: Visualization **(7)**, included in the third instance, of the Genome Domain.

Another issue, still tied to figure 6.11, is how limited the system is regarding marks, colors, and general encoding channels. These small details directly affect the creation of complex visualizations, such as the one we hoped we could achieve when answering this question. This happens because any sizable number of unique values is too much for the system to handle without performing encoding repetitions, which renders the visualization unintelligible. Another issue is the fact that the visualization is scaled up in terms of height but not in terms of width, which contributes to the lack of readability. The inability to change axis size/range handicaps possible visualizations that would benefit

from having more space to help with cluttering. These cases are widespread within data visualization, directly hindering our approach, expectations, and results.

6.2.2 Business Domain

The domain has a focal point in sales, more specifically the sales that occur in a European superstore, and the information regarding each sale/order (i.e. Order Date, Category, Discount, Profit, etc...). The Business dataset [114], is a fully functioning dataset that expresses orders from customers and their respective information.

Figure 6.12 expresses the business domain, the current domain question, the respective abstract question, and analytical tasks. The instances represent two testing alternatives, with different angles to approach the same question.

Business Domain	
Domain Question	What is the average discount per sale, for each month of the last years, and how do they compare ?
Abstract Question	The user wants to know the differences between the average discount per month.
Analytical Task Decomposition	**Compute Attribute** (Attribute with Month and Year); **Aggregate** (Average/Mean); **Compare**
First Instance - Composed of the two data fields identified in the domain question, Discount and MonthYear. These data fields are described as linear, for Discount, and temporal for MonthYear. An aggregation (mean) will be applied to Discount. Both attributes were declared without an assigned axis.	
Second Instance - The MonthYear attribute was changed to the Month attribute, where only the month is displayed independently of the year, to create a different notion of temporal scale, in this case, it would be a cyclical one. This creates a new approach to answer the domain question and yield visualizations.	

Figure 6.12: Summary table relative to the **business domain**, where the domain question, abstract question, analytical tasks, and domain instances are present.

The determined domain question was **"What is the average discount per sale, for each month of the last years, and how do they compare ?"**. In this dataset, as a consequence of the number of data fields, the only attributes worthy of mention are **Discount**, which is the percentage of discount of a specific order, and **Order Date**, which represents the specific date of an order.

Evaluating the domain question draws the necessary analytical tasks as **Compute Attribute** [Derive], **Aggregate** [Derive], and **Compare**. With the more well-rounded system, there would not be the need to have a compute attribute, but the absence of a good way to manipulate dates creates the need to specify a new attribute that describes the months and years of each order, that will be named **MonthYear**. The other task, **Compare**, is easily extracted from the original domain question.

First instance relates to a characterization of Discount and MonthYear, the two required data fields to answer the domain question. Discount follows a continuous type while MonthYear has a temporal data type. The domain question nature requires that discount

must appear as the average for each month of a year. Both Discount and MonthYear were not mapped to any axis/encoding channel.

- **(1)** - Represents the aggregate field Discount, in the y-axis with a linear scale, and MonthYear is placed in the x-axis, with a similar linear scale. The resulting visualization is competent enough to answer the question, even though it might not be easy to visualize quickly because the individual marks are close, which can confuse the observer.

- **(2)** - Switches the data field layout between the x-axis and y-axis, when compared to (1). In terms of answering the domain question, it might have a slight additional reflection on the knowledge because the horizontal disposition helps with the comparison.

- **(3)** - Composes a bar chart, frequently regarded as very effective in comparison tasks, with MonthYear changing from a linear scale to a discrete one, in the y-axis, and Discount on the x-axis with a linear scale. This change helps with both accuracy and comparative easiness. This visualization is the best-performing one within this instance.

Business **visualization (3)**, present in Figure 6.13, is the best visualization since it follows general rules of thumb for good visualization that wish to compare values.

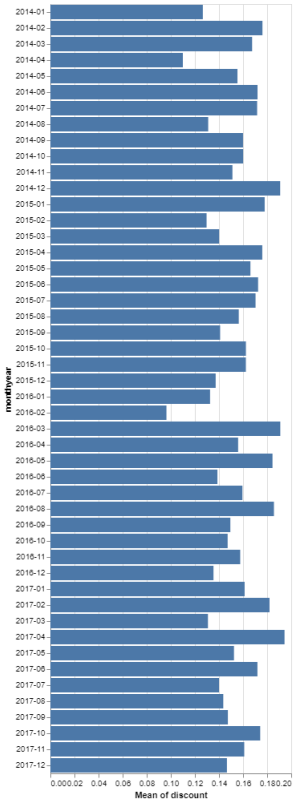

Figure 6.13: Visualization **(3)** from Business Domain, obtained in the first instance.

Bar charts are great at comparing, the data types are correctly addressed, the months of each year are correctly ordered, and the original size makes the visualization easily readable.

In the **second instance** there is a change from MonthYear, a field that has each month associated with a specific year (e.g. 01-2017), to **Month** where there is a single representation of the months regardless of the year (e.g. Jan). This change was made so that a divergent notion of temporal scales could be used to analyze this question. This means that the problem is turned from a linear scale to a cyclical scale, while the remaining properties are mirrored from the **first instance**. Our objective was to achieve a new way to tackle the original question and obtain new visualizations that might be helpful, likewise:

- **(4)** - Is a bar chart with Month on the x-axis, as a discrete variable, and the aggregated Discount as a linear variable on the y-axis. The visualization is satisfactory and makes the values easy to be compared. It could be slightly better if it featured horizontal bars.
- **(5)** - Generated a visualization very similar to (4), where the only tangible change is the mark. Instead of bars, this visualization has a line for a mark. In practice, the difference is in terms of adequacy the difference is not substantial but this visualization is worse when compared to (4).
- **(6)** - The last visualization of this domain continues the trend set by (5) and changes the mark. In this case, the line mark is changed to circles like in a bubble chart.

Figure 6.14 shows **visualization (4)** suitable to answer the domain question. Its strong points come from being concise, displaying a good mark for comparing and having good scaling.

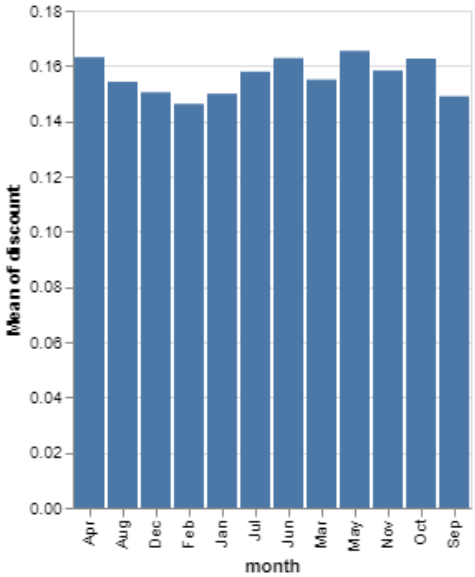

Figure 6.14: Visualization **(4)** from the business domain, obtained in the second instance.

This visualization could approach the quality of Visualization **(3)** if it ordered the months in their calendar order, instead of alphabetically. This happens because in this visualization, unlike Figure 6.13, months are strings, e.g. April, instead of numbers, e.g. 2015-01.

Two negative aspects withdrawn from this visualization are the system's incapability of manipulating data and the need to adjust data types because of a lack of default support. The second point relates to including, in this case, extra analytical tasks like **Compute Attribute**, to derive the month from the existing Order-Date field in the dataset.

6.2.3 Spatial Domain

The domain is reflected by the dataset containing information relative to different countries' cities and their latitude and longitude in the five hundred most populated cities in the world, population is also part of the data. Our focus on spatial data will be carried out through **lat** and **lng**.

The spatial domain specifications, foreseen in Figure 6.15, express a special type of overview of our approach because it focuses on a type of data, that when studied within the data visualization field, has some particular properties, due to its intrinsic nature.

Spatial Domain	
Domain Question	Can a connection be set between a city's coordinates and a city's population size ?
Abstract Question	The user wants to know the relationship between the city location and its population.
Analytical Task Decomposition	**Relate** (Between lat, lng, and population)
First Instance - Classifies the visualization by defining the three identifed data fields. Latitude (lat) is a continuous field, Longitude (lng) is also a continuous field, and Population (population) is continuous as well. There is no mapping relative to axis or encodings.	
Second Instance - Follows the real-world abstraction with the x-axis and y-axis being defined as latitude and longitude. Once more, the field population is free to be manipulated by the recommendation system.	

Figure 6.15: Summary table with details relative to the **spatial domain**, where the domain question, abstract question, analytical tasks, and domain instances are present.

The proposed domain question was **"Can a connection be set between a city's coordinates and a city's population size?"**. Within the question, there is a clear sense to relate a city's latitude and longitude with its population. All the information is already set in the dataset so the only requisite analytical task is the proper visualization-generating task **Relate**. This means that the data fields at hand to answer the domain question are the city's latitude (**lat**), longitude (**lng**), and population (**population**).

Spatial's domain **first instance** exhibits the visualizations generated through the usage of the previously identified data fields: lat, lng, and population. All these fields are maintained as continuous fields with linear scales. No field is explicitly assigned to an

axis and/or encoding. Thus, the recommendation system will create them as it deems necessary, which results in the subsequent visualizations:

- **(1)** - Latitude (lat) is set on the x-axis and Longitude (lng) comes stated as the y-axis, both as linear scale axes. The population is mapped to the color encoding where the intensity of the color, indicates a higher or lower populated city. The scatterplot is correct, but the order of the axes does not allow for a good display of information.
- **(2)** - It mirrors visualization (1) but exchanges the data field's arrangement. Population on to the y-axis, lng on the x-axis, and lat with the color encoding. This specific visualization is even worse since it is a struggle to be able to relate lat and lng to their respective population.
- **(3)** - Similar to (2), with the only change happening between lat and lng, where lat switches to the x-axis and lng gets mapped to the color encoding. It continues being a visualization that is way below expectations.

In this first instance, the three top-performing visualizations were unsuccessful in their task of providing a good result to answer the domain question. The recommendation system has minimal default data types, and the non-existence of geospatial data, e.g. coordinates, negatively impacts these visualizations because the system is unable to weigh intrinsic differences between fields like population and lat or lng. Figure 6.16, exemplifies this perfectly when the system allocates lng to the x-axis but maps lat to the color encoding, leaving population as the y-axis.

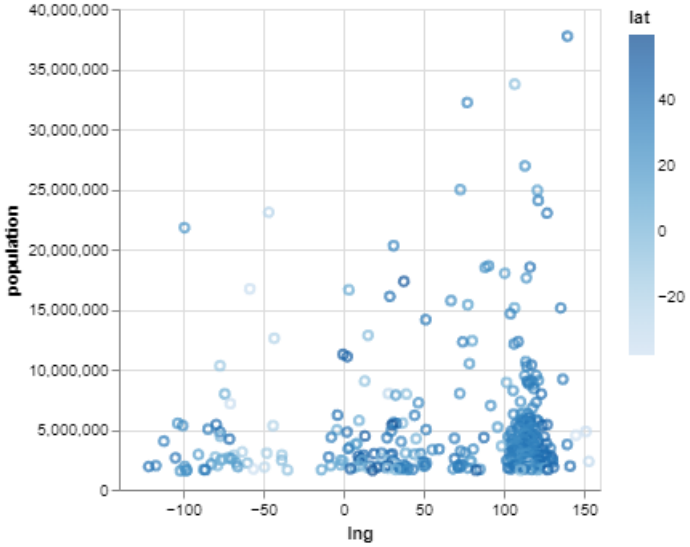

Figure 6.16: Visualization **(2)** from Spatial Domain, obtained in the first instance.

Second instance sets up and applies latitude (lat) as a continuous field in the x-axis, longitude (lng) as a continuous field in the y-axis, and population is given as a wildcard. The belief is that this instance will produce undeniably better visualizations since it follows the real-world applications of the lat and lng variables.

- **(4)** - Illustrates the best possible visualization within all examples. Longitude and Latitude are easy to relate to each other. Besides, using the population field in the color encoding makes it even identify possible clusters of cities (**Find Clusters** could have been a good analytical task to answer this domain question).

- **(5)** - Demonstrates a bad visualization that comes as a result of the system's bias towards aggregating continuous values. This visualization represents all data fields as an aggregate sum function. The X-axis, y-axis, and size encoding, were mapped to lat, lng, and population, respectively.

- **(6)** - Renders another bad visualization that is very similar to (5) and suffers from the aggregating bias. The only difference is that population represents the color encoding as an alternative to size.

Figure 6.17, depicts the possible best visualization within the system's limitations. The data fields and their organization within the visualization space follow the real-world abstraction with the x-axis and y-axis being defined by latitude (lat) and longitude (lng). Besides, the system set the population field to be mapped to the color encoding, where the amount of population is associated with color intensity. The visualization offers good answering capabilities since the system provided a positive solution for the inclusion of the population field. However, the visualization lacks the necessary treatment for this data type. Commonly, in data visualization, the usage of spatial data is paired with necessary projections and utilization of maps-like background images, an important abstraction layer, as a way to ease the users' interpretation of coordinates and scales. Compared to Figure 6.16 it is possible to see how meaningful it is to order fields, within a visualization space, correctly.

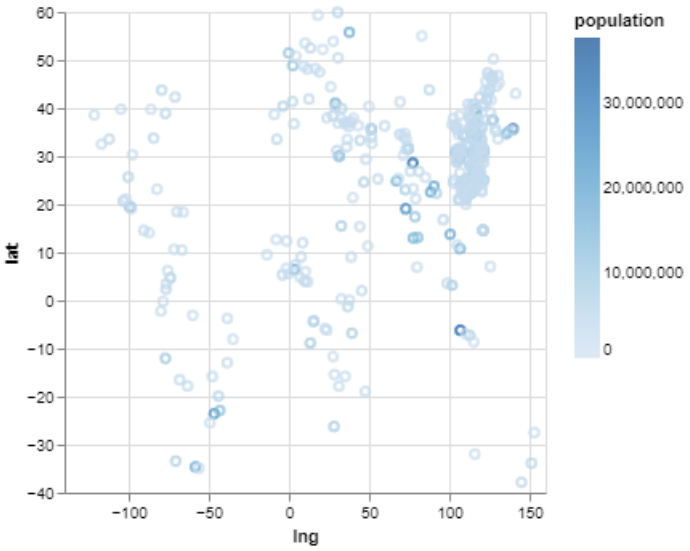

Figure 6.17: Visualization **(4)** from Spatial Domain, obtained in the second instance.

6.2.4 Visualization Conclusions

In summary, this section reflects upon some of the results achieved through the recommendation system when answering different domain questions. We specified an example per each instance of domain, that allowed us to highlight some of the strengths and weaknesses within the system and the implemented modifications. From the study of all testing examples we understood the following:

- **Simple Visualization** are typically **well-executed** by the system and consequently by the analytical tasks. Whereas, **Complex Visualizations** with multiple data fields **struggle** with the low quantity of available mapping options and display space.
- **Lack of system support** for **specific data type** transformation and manipulation. Additionally, there is also a **shortage of configure/setting options** that allows changes to the visualization, display size, and graphical primitives.
- **Correct execution of visualizations based on analytical tasks**. Nonetheless, the constraints did **not address some issues** related to the inherent system's preference for aggregating data and allocating data fields to different encodings.
- **Shortage** of available charts and existing encodings, **especially compared to other tools**.
- The system **must always incorporate a single visualization-generating analytical task**, to produce visualizations. However, domain questions might necessitate more than one, or not need one at all, but the recommendation system forces this behavior and therefore is another known limitation.
- **Testing in instances** enables the examination of how specific characteristics **influence the system evaluation** and its **recorded outputs**.

6.3 Comprehensive Example

In this section, the major objective is to thoroughly exemplify a **start-to-finish** functioning example of our approach. Starting with the choice of a domain and the respective domain question, and concluding with the generated visualizations. Thus, no additional motivation exists to scrutinize the elaborated example.

Considering this, the example will employ an already evaluated domain, domain question, and translation of the domain question. Complementary, the chosen domain was business/sales since it had the best domain question translation success ratio when it came to our results in **LLMs** testing. To maintain this cohesion, one of the best-ranked models in our testing (GPT-4) was also selected to feature in this demonstration.

Since our approach starts by evaluating conceptual models, it is paramount to initiate our methodology as such. Figure 6.18 exemplifies a possible conceptual model, in this case,

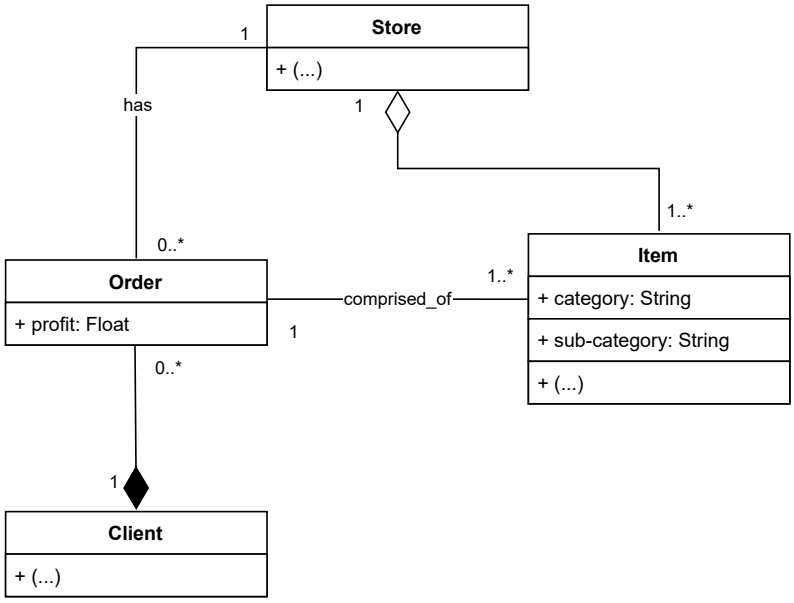

Figure 6.18: Example of a Business/Sales domain conceptual model. Inspired by Tableau's Superstore dataset [114].

it was simplified to show only the necessary data fields to answer the domain question. The model represents how a store manages its inventory, and orders from clients.

When evaluated by our meta-model, conceptual models are assessed over data mapping, analytical tasks, and visualizations. Data fields associated with classes are mapped to their respective dataset counterparts, this mapping then enables the correct characterization of data fields into their respective data structure and type. The domain question and respective analytical tasks, in this case, the first is defined by the user, and the latter is determined by the LLM. When choosing the domain question, keeping a low complexity and selecting a relatively common/normal question was preferred.

To serve this purpose, the already tested domain question **"What is the total profit for each sub-category of products?"** was picked. Answering such a question correctly was initially classified, in our testing, as only needing the analytical task ***Aggregate*** with a sum function of the field **Profit** grouped by **Sub-Category**. Although this would be enough, the recommendation system's nature requires the presence of a single visualization-generating task. Thus, it was necessary to ask the model to include one of these tasks as an addition to the original, and GPT-4 appointed ***Compare*** to be the extra task. Afterward, to ensure continuity, another choice is the order of the attributes and which data fields will be shown in the visualization display. The LLM decision was to designate the discrete values (Sub-Category) into the x-axis and have the aggregation of the continuous one (Profit) in the y-axis.

With all the available information regarding the **domain, analytical tasks, and design choices it is possible to obtain the resulting visualizations**.

This information must be processed into the data repository for this to happen. Inside the data repository the crucial information, related to the analytical tasks is kept in order.

First, the supportive analytical tasks, then the visualization-generating tasks, as seen in Listings 6.1 and 6.2.

```
{
        questionID: 1413,
        taskID: 14131,
        task_selection: [{
                        task: "Derive",
                        subtask: [{
                                        task: "Aggregate",
                                        type: "sum",
                                        attribute: "profit",
                                        groupby: "subcategory"}],
                        domain_specification: "Aggregate the values of profit
                            ↪ grouped by their respective subcategory.",
                        analytical_specification: "What values of profit are
                            ↪ achieved in each subcategory?",
                        expected_output: "A dataset with the list of aggregate
                            ↪ values of profit for unique subcategories."
                }
        ]
}
```

Listing 6.1: Representation of this domain's analytical task **Aggregate** in the data repository.

```
{
        questionID: 1413,
        taskID: 14132,
        task_selection: [{
                        task: "Compare",
                        subtask:[],
                        domain_specification: "Compare the total profit of the
                            ↪ different sub-categories.",
                        analytical_specification: "What differences (when
                            ↪ comparing) are found in the profits between
                            ↪ sub-categories.",
                        expected_output: "A dataset with the list of total profit
                            ↪ values for each of the three different
                            ↪ sub-categories"}
        ]
}
```

Listing 6.2: Representation of this domain's analytical task **Compare** in the data repository.

After setting up the necessary environment, the data repository will be analyzed by the parser, which will manipulate and organize the input, and assimilate a specific-formatted file (Listing 6.3) to be handled by the ASP.

```
attribute(task,root,compare).
attribute(number_rows,root,9994).

entity(field,root,(f,0)).
attribute((field,name),(f,0),subcategory).
attribute((field,type),(f,0),string).
attribute((field,unique),(f,0),17).

entity(field,root,(f,1)).
attribute((field,name),(f,1),profit).
attribute((field,type),(f,1),number).
attribute((field,unique),(f,1),7287).

entity(view,root,(v,0)).
entity(mark,(v,0),(m,0)).

entity(encoding,(m,0),(e,0)).
attribute((encoding,field),(e,0),subcategory).
attribute((encoding,channel),(e,0),x).

entity(encoding,(m,0),(e,1)).
attribute((encoding,field),(e,1),profit).
attribute((encoding,channel),(e,1),y).
attribute((encoding,aggregate),(e,1),sum).

entity(scale,(v,0),(s,0)).
attribute((scale,channel),(s,0),x).
attribute((scale,type),(s,0),ordinal).

entity(scale,(v,0),(s,1)).
attribute((scale,channel),(s,1),y).
attribute((scale,type),(s,1),linear).
```

Listing 6.3: Visualization file responsible for characterizing the visualization within the recommendation system.

Lastly, the recommendation system takes the file and starts the parsing and translation process, once this is over the data is mapped and encoded into the engine, and the output visualization(s) are rendered for the user to analyze them. These are exhibited in Figure 6.19 and 6.20.

In this specific instance, and as a consequence of the created constraints (both soft and hard) the model adapts its visualization generation to the ***Compare*** analytical task and the usage of a bar chart, line chart, and a radial chart as the most valuable answers. Visualizations present in Figure 6.19 are reliable and would most likely serve a user with a well-versed response, although the bar chart visualization would be typically regarded

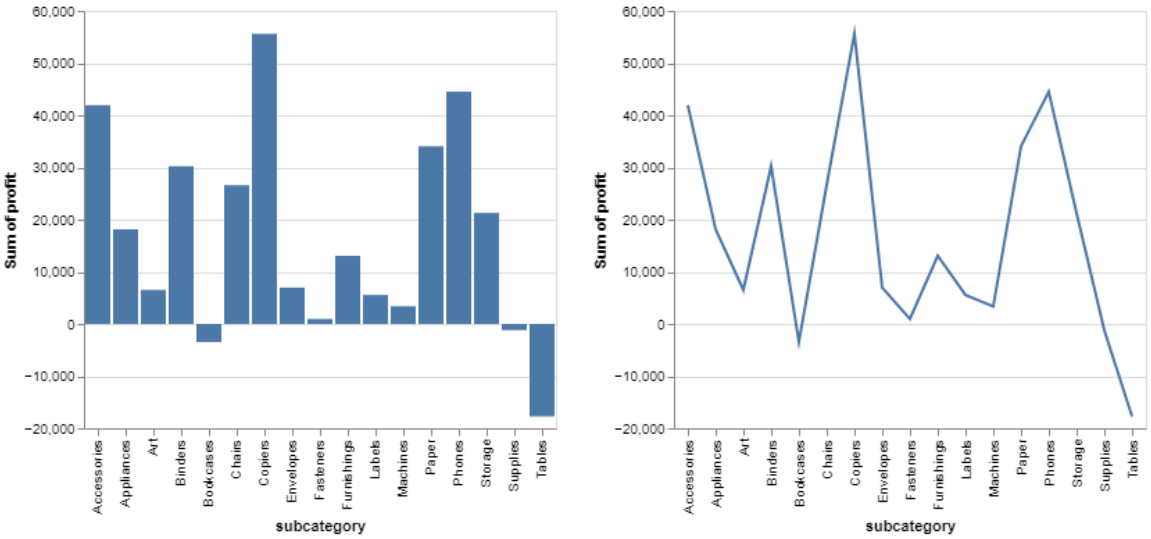

Figure 6.19: Best resulting visualizations, 1st and 2nd ranked from left-to-right, obtained by the recommendation system to answer the domain question: **What is the total profit for each sub-category of products?**

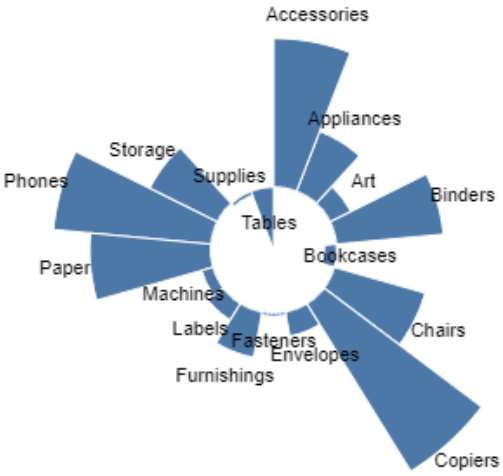

Figure 6.20: Third-best visualization, from the recommendation system to answer the domain question: **What is the total profit for each sub-category of products?**

as the best one. On the contrary, Figure 6.20 is a worse visualization since it is difficult to perform any type of comparison with precision, due to the lack of information regarding the sum of profits. Following our approach, originally seen in Figure 1.1, the next step would be to analyze the results and understand if the user was satisfied and capable of answering their initial question or if they had any more questions to be inputted to the system, which supports the idea behind the iterative process to solve the user requirements.

7

Conclusions

7.1 Overview & Discussion

This dissertation focused mainly on investigating and researching how to prove our initial hypothesis surrounding a model-driven approach to creating data visualizations. Our conjuncture had a primary objective of being proved through an ontologically correct meta-model, as the core foundation, that would conduct information to other systems/technologies.

Due to the theoretical nature of this dissertation, there was a great extent of needed research. Leveraging knowledge regarding modeling, data analysis, data visualization principles, and large language models functioning. **Research emerged firstly through the need to understand basic mechanisms** regarding data analysis, data visualization, and the conceptual modeling capacities to acknowledge the potential lengths of our approach and, simultaneously, its limitations.

The first advancements came as a result of **expanding and developing the existing meta-model**. This expansion was continuously done throughout the dissertation, by using an iterative process, since it is our approach's foundation. Therefore, it must be kept updated and trying to achieve the highest level of completeness.

After achieving a stable version of the meta-model, it was necessary to initiate a study about domain questions, generic domain questions, analytical tasks, and taxonomies. This study revolved around a **survey of taxonomy and typology literature, in order to create a personalized taxonomy proposal**. This proposal would come to be executed by our model, translation technologies, and visualization system. It resulted in a multi-purpose set of tasks able to tackle a significant amount of domain questions and their different interpretations, as can be seen throughout the different examples/tests executed in this thesis.

With all this information it was possible to start **performing different experiments regarding translation: data expert translations and language model translations**. In data expert translations, the objective was to test the taxonomy and provide examples that would allow for the drawing of a baseline that characterized the taxonomy's effectiveness

with certain questions and/or domains. Looking at translations made by large language models, the popular state-of-the-art natural language processing models were tested as a possibility to recognize the effectiveness and quality of these technologies when working within constricted domains, with the intent to directly obtain sets of analytical tasks stemming from the processing of preliminary domain questions.

Thereafter, a **recommendation visualization system was chosen** and extended in various concepts. The system was modeled to incorporate our proposed taxonomy, additional data types, and execute sets of analytical tasks to achieve arrays of visualizations, through the usage of an external program.

The **testing and evaluation phase took into practice the foundations directly associated with this thesis**, it is comprised of three prime testing stages where the focus, on two of the tests, is on individually identifying advantages and drawbacks directly correlated with the utilization of a technology and how does it link to our approach. In the remaining test, a full-on start-to-finish examination was conducted to provide insight into the diverse components and how they work together to achieve results. Inevitably, this test was primarily done to showcase the pipeline and needed logical reasoning behind creating a single visualization.

Nevertheless, testing for **Translation** characterized that LLM technologies have the potential to become relevant within the scope of data analysis and data visualization due to their data and natural language processing proficiency. Experiments first came to light with the idea of using popular state-of-the-art language models in our translation process to directly obtain sets of analytical tasks from domain questions. Our results mirror this prospect by providing positive - although not extraordinary - success rates in smaller-sized samples even without extracting the most, theoretically speaking, from these language models.

Visualization testing, amounted to showing some of the recommendation system answers and simultaneously judging their viability, accuracy, and effectiveness. This testing came about as a step forward to understand the generated visualizations and how our approach influences the creation of visualization. All this is possible due to how we implemented our approach using models as a knowledge base to build visualizations.

Overall, the core analysis regarding the last chapter is an assured possibility of implementing and effectively achieving our hypothesis, of developing a model-driven data visualization approach. The basis on which the approach is built upon, the creation of knowledge through models, and **its usage as an ontologically sound foundation is proven to work and be correct**, as it can be seen by the analyzed examples throughout this dissertation.

However, this approach alone is not enough to create visualizations and requires other systems to provide direct support. Systemic strengths and weaknesses will result from the utilization of said systems, which might not always correlate with the approach's pros and cons. Although there are limitations to the supporting systems, such as the recommendation system, our consensus is that the approach is broad enough to not be

limited, in most cases, by the constraints imposed by these other systems. Finding or developing systems that do not bottleneck the methodology is the key to unlocking its full potential.

7.2 Model-Driven Analytics in the area of LLMs ?

Within Model-Driven approaches, there is an overlap that reinforces a direct correlation between model and knowledge-driven approaches. This happens since knowledge-driven methodologies, tend to base themselves on ontologies that manage the system's requirements, where ambiguity, inconsistency, and incompleteness are minimized [13]. Furthermore, in data analytics, knowledge-driven designs take advantage of knowledge repositories that represent different analytical concepts [56], e.g. requirements, datasets, and decision-making processes. In our approach, we converge on the usage of knowledge repositories such as our meta-model, taxonomy, and external knowledge regarding visualizations in our recommendation system.

Works integrating LLMs are currently at an all-time high since interest in this technology has spread into varying domains. Literature works regarding analytics, and the usage of LLMs, have not been widely featured in the sub-field of model-driven approaches. However, steps toward incorporating ontologies and construction of knowledge-graphs [75] with some degree of success are being achieved. This can be used as an argument to support our belief that model-driven analytics might have reinforced interest within the LLM research space. For example, from helping users build accurate conceptual models, providing constructive insights within the data, serving as a potential helping hand in domain question analysis to both users and experts, *et cetera*.

These tasks are simultaneously related to expanding some areas of work regarding language models and could serve as a core point not only to modeling and model-driven approaches but also to other fields.

In summary, LLM-related research skyrocketed and this statement alone leads to many questions surrounding its usability within specified domains of interest. We hypothesize that model-driven analytics might have an interest in this field. This stems from the fact that it can benefit greatly from automation and knowledge-expanding perspectives, while simultaneously sharing multiple common points with different fields of study.

7.3 Future Work & Analysis

Although strides were made throughout the development of this dissertation, some challenges were faced that were functionally out of the scope of this thesis or not possible to properly execute, within the given time frame. The correct definition of this future work shows that our approach can benefit from multiple improvements that might be crucial to further developing this methodology.

7.3.1 Meta-Model Extension

Exploring new features regarding the meta-model is a possible topic of future work. Inclusion of further data structures, possible impacting ways of organizing and setting analytical tasks, and having in-depth data visualization analysis components beyond the analytical tasks and the data visualizations.

Especially when judging the inclusion of possible ways for both the user and expert to interact with the model and the resulting visualizations, could enable a whole new layer of abstraction that would test and evaluate our approach with individuals outside of the field of data visualization.

7.3.2 Taxonomy Enrichment

As mentioned in the taxonomy chapter, 3, currently our taxonomy proposal supports a limited number of operations that we [the Team] deemed as necessary for the analysis of data. Looking at the possibility of future work, this might entail adding new analytical tasks - for specific analysis (e.g. Search, Summarize) or related to specific data types (e.g. Scaling, in spatial data) - or perhaps including a new abstraction layer by enabling visualization interaction (e.g. Overview [108], Configure [116]), which we did not include, as a strategical decision, to keep the taxonomy simple, although complete, to test our approach.

This would create a more complete taxonomy with a bigger user-oriented focal point, which could contribute towards testing with users, in a user-friendly environment.

7.3.3 Recommendation System

The visualization recommendation system, Draco, proved a great tool in this dissertation since it drew in its system multiple features that we were looking for within the data visualization sphere. However, the work done on the system was mostly through the lens of following our foundational guidelines.

Future work could explore means to expand the system's visualization options, by extending the ASP to enable features present in the engine (Vega-Altair) that are not currently in the recommendation system. Achieving better visualization results with a bigger training dataset, more specifically, with a larger amount of visualizations applied to varying domains with different characteristics.

Such changes would attempt to improve two of the main issues present in the current system, by increasing the visual options and the accuracy of design choices for each analytical task. The overall quality and effectiveness of visualizations might improve with these additions.

Also designing and building a dataset for the analytical tasks **Find Patterns**, **Find Extremes**, **Find Clusters**, and **Find Anomalies** would be important to provide consistency to the model and its choices, when generating visualizations dependent on these tasks.

7.3.4 LLM Fine-Tuning

Concerning the large language models component, the biggest possible improvement, compared to the tests that were executed in this thesis evaluation, is the chance to produce a generative model focused solemnly on the translation of domain questions into analytical questions. This means the possibility of fine-tuning an LLM to achieve a model catered to our specific task at hand. Nevertheless, the need to fine-tune a model from scratch would be a laborious task from the data collection and quality standpoint because this technique not only requires considerable amounts of data but also a whole methodology that must be adopted in order to accurately shape the model [111].

This type of future work could be interesting not only to achieve a higher refinement in analytical task translation but also in the field of data analysis and data visualization since it could help bridge the gap between regular individuals and data experts.

Bibliography

[1] K. K. A. Parameswaran. *Zenvisage*. Accessed on Fev. 2, 2024. 2019. URL: https://zenvisage.github.io/ (cit. on p. 30).

[2] A. Abela. *Advanced Presentations by Design: Creating Communication that Drives Action*. Pfeiffer essential resources for training and HR professionals. Wiley, 2008. ISBN: 9780470378359. URL: https://books.google.pt/books?id=z2S0Fz_gD2wC (cit. on pp. 43, 45–47, 49, 55).

[3] D. Agrawal et al. "Challenges and Opportunities with Big Data 2011-1". In: 2011 (cit. on p. 8).

[4] R. Aisami. "Learning Styles and Visual Literacy for Learning and Performance". In: *Procedia - Social and Behavioral Sciences* 176 (2015-02), pp. 538–545. DOI: 10.1016/j.sbspro.2015.01.508 (cit. on p. 22).

[5] D. Alexandre and J. Tavares. "Introduction of Human Perception in Visualization". In: *International Journal of Imaging* 4 (2010-01) (cit. on p. 10).

[6] A. C. P. Almeida. "Abordagem baseada em Modelos para Suporte à Visualização Analítica". Unpublished thesis. 2024 (cit. on p. 64).

[7] B. Alper et al. "Visualization Literacy at Elementary School". In: *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. CHI '17. Denver, Colorado, USA: Association for Computing Machinery, 2017, pp. 5485–5497. ISBN: 9781450346559. DOI: 10.1145/3025453.3025877. URL: https://doi.org/10.1145/3025453.3025877 (cit. on p. 22).

[8] R. Amar, J. Eagan, and J. Stasko. "Low-Level Components of Analytic Activity in Information Visualization". In: *Proceedings of the Proceedings of the 2005 IEEE Symposium on Information Visualization*. INFOVIS '05. USA: IEEE Computer Society, 2005, p. 15. ISBN: 078039464x. DOI: 10.1109/INFOVIS.2005.24. URL: https://doi.org/10.1109/INFOVIS.2005.24 (cit. on pp. 39–41, 46, 47, 49, 52).

[9] D. Ameller. "Considering Non-Functional Requirements in Model-Driven Engineering". MA thesis. 2011-01 (cit. on p. 32).

[10] A. S. Association. *Dataset repository before 1993*. https://community.amstat.org/jointscsg-section/dataexpo/dataexpobefore1993. Accessed: 2024-02-28. 2024 (cit. on p. 75).

[11] C. Atkinson and T. Kühne. "Kühne, T.: Model-driven development: a metamodeling foundation. IEEE Software 20(5), 36-41". In: *Software, IEEE* 20 (2003-10), pp. 36–41. DOI: 10.1109/MS.2003.1231149 (cit. on p. 32).

[12] K. D. Bailey. "Typologies and taxonomies: An introduction to classification techniques." In: 1994 (cit. on p. 18).

[13] M. Bandara, F. A. Rabhi, and M. Bano. "A knowledge-driven approach for designing data analytics platforms". In: *Requirements Engineering* 28.2 (2023), pp. 195–212. DOI: 10.1007/s00766-022-00385-5. URL: https://doi.org/10.1007/s00766-022-00385-5 (cit. on p. 108).

[14] J. Bertin. *Semiology of Graphics*. University of Wisconsin Press, 1983. ISBN: 0299090604 (cit. on pp. 14, 44).

[15] A. Bordes et al. "Translating Embeddings for Modeling Multi-relational Data". In: *Advances in Neural Information Processing Systems*. Ed. by C. Burges et al. Vol. 26. Curran Associates, Inc., 2013. URL: https://proceedings.neurips.cc/paper_files/paper/2013/file/1cecc7a77928ca8133fa24680a88d2f9-Paper.pdf (cit. on p. 26).

[16] K. Börner et al. "Data visualization literacy: Definitions, conceptual frameworks, exercises, and assessments". In: *Proceedings of the National Academy of Sciences of the United States of America* (2019). DOI: 10.1073/pnas.1807180116 (cit. on pp. 39, 44–47, 49, 52).

[17] K. Börner et al. "Investigating aspects of data visualization literacy using 20 information visualizations and 273 science museum visitors". In: *Information Visualization* 15.3 (2016), pp. 198–213. DOI: 10.1177/1473871615594652. eprint: https://doi.org/10.1177/1473871615594652. URL: https://doi.org/10.1177/1473871615594652 (cit. on p. 22).

[18] J. Boy et al. "A Principled Way of Assessing Visualization Literacy". In: *IEEE Transactions on Visualization and Computer Graphics* 20.12 (2014), pp. 1963–1972. DOI: 10.1109/TVCG.2014.2346984 (cit. on p. 22).

[19] M. Brehmer and T. Munzner. "A Multi-Level Typology of Abstract Visualization Tasks". In: *IEEE Transactions on Visualization and Computer Graphics* 19.12 (2013), pp. 2376–2385. DOI: 10.1109/TVCG.2013.124 (cit. on p. 19).

[20] T. B. Brown et al. *Language Models are Few-Shot Learners*. 2020. arXiv: 2005.14165 [cs.CL] (cit. on pp. 53, 82, 84).

[21] P. Brusilovsky. "Methods and Techniques of Adaptive Hypermedia". In: *User Modeling and User-Adapted Interaction* 6 (1996-07), pp. 87–129. DOI: 10.1007/BF00143964 (cit. on p. 55).

[22] S. Card, J. Mackinlay, and B. Shneiderman. *Readings in Information Visualization: Using Vision To Think*. 1999-01. ISBN: 978-1-55860-533-6 (cit. on p. 11).

[23] E. Chi. "A taxonomy of visualization techniques using the data state reference model". In: *IEEE Symposium on Information Visualization 2000. INFOVIS 2000. Proceedings*. 2000, pp. 69–75. DOI: 10.1109/INFVIS.2000.885092 (cit. on p. 11).

[24] W. Cleveland and R. Mcgill. "Graphical Perception and Graphical Methods for Analyzing Scientific Data". In: *Science (New York, N.Y.)* 229 (1985-09), pp. 828–33. DOI: 10.1126/science.229.4716.828 (cit. on p. 15).

[25] S. Conn and L. Forrester. "Model Driven Architecture: A Research Review for Information Systems Educators Teaching Software Development". In: *Number* 4 (2006-08) (cit. on p. 33).

[26] W. Cui. "Visual Analytics: A Comprehensive Overview". In: *IEEE Access* 7 (2019), pp. 81555–81573. DOI: 10.1109/ACCESS.2019.2923736 (cit. on pp. 1, 7, 10, 21).

[27] A. Dahanayake, J. Huiskonen, and Y. Kiyoki. *Information Modelling and Knowledge Bases XXXI*. Frontiers in Artificial Intelligence and Applications. IOS Press, 2020. ISBN: 9781643680453. URL: https://books.google.pt/books?id=4mTIDwAAQBAJ (cit. on p. 35).

[28] M. Dastani. "The Role of Visual Perception in Data Visualization". In: *Journal of Visual Languages & Computing* 13.6 (2002), pp. 601–622. ISSN: 1045-926X. DOI: https://doi.org/10.1006/jvlc.2002.0235. URL: https://www.sciencedirect.com/science/article/pii/S1045926X02902351 (cit. on p. 16).

[29] L. Delcambre et al. *A Reference Framework for Conceptual Modeling: Focusing on Conceptual Modeling Research*. 2018-11. DOI: 10.13140/RG.2.2.33041.07521 (cit. on pp. 37, 62).

[30] L. M. L. Delcambre et al. "A Reference Framework for Conceptual Modeling". In: *Conceptual Modeling*. Ed. by J. C. Trujillo et al. Cham: Springer International Publishing, 2018, pp. 27–42 (cit. on p. 37).

[31] Ç. Demiralp et al. "Visual Embedding: A Model for Visualization". In: *IEEE Computer Graphics and Applications* (2014). DOI: 10.1109/mcg.2014.18 (cit. on p. 11).

[32] J. Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2019. arXiv: 1810.04805 [cs.CL] (cit. on p. 53).

[33] V. Dibia and Ç. Demiralp. *Data2Vis: Automatic Generation of Data Visualizations Using Sequence to Sequence Recurrent Neural Networks*. 2018. arXiv: 1804.03126 [cs.HC] (cit. on p. 25).

[34] M. Ferreira de Oliveira and H. Levkowitz. "From visual data exploration to visual data mining: a survey". In: *IEEE Transactions on Visualization and Computer Graphics* 9.3 (2003), pp. 378–394. DOI: 10.1109/TVCG.2003.1207445 (cit. on p. 10).

[35] S. Few. *Show Me the Numbers: Designing Tables and Graphs to Enlighten*. Analytics Press, 2012. ISBN: 9780970601971. URL: https://books.google.pt/books?id=1xiHLgEACAAJ (cit. on pp. 43–47, 49, 52, 55).

[36] S. L. Franconeri et al. "The Science of Visual Data Communication: What Works". In: *Psychological Science in the Public Interest* 22 (2021), pp. 110–161 (cit. on pp. 9, 10, 12, 13).

[37] A. Gandomi and M. Haider. "Beyond the hype: Big data concepts, methods, and analytics". In: *International Journal of Information Management* 35 (2015-04), pp. 137–144. DOI: 10.1016/j.ijinfomgt.2014.10.007 (cit. on p. 8).

[38] M. Gebser et al. "Multi-shot ASP solving with clingo". In: *CoRR* abs/1705.09811 (2017) (cit. on pp. 27, 66, 68).

[39] A. GELMAN. "Exploratory Data Analysis for Complex Models". In: *Journal of Computational and Graphical Statistics* 13 (2012-01). DOI: 10.1198/106186004X11435 (cit. on p. 7).

[40] A. Gogolou et al. "Comparing Similarity Perception in Time Series Visualizations". In: *IEEE Transactions on Visualization and Computer Graphics* 25.1 (2019), pp. 523–533. DOI: 10.1109/TVCG.2018.2865077 (cit. on p. 56).

[41] Google. *Gemini*. https://gemini.google.com/u/2/app. Accessed: 2024-02-17. 2024 (cit. on p. 84).

[42] N. Guarino, G. Guizzardi, and J. Mylopoulos. "On the Philosophical Foundations of Conceptual Models". In: 2019-08 (cit. on p. 34).

[43] G. Guizzardi, H. Herre, and G. Wagner. "On the General Ontological Foundations of Conceptual Modeling". In: 2002-09. ISBN: 978-3-540-44277-6. DOI: 10.1007/3-540-45816-6_15 (cit. on p. 35).

[44] S. Haroz and D. Whitney. "How Capacity Limits of Attention Influence Information Visualization Effectiveness". In: *IEEE Transactions on Visualization and Computer Graphics* 18.12 (2012), pp. 2402–2410. DOI: 10.1109/TVCG.2012.233 (cit. on p. 55).

[45] R. Harris. *Information Graphics: A Comprehensive Illustrated Reference*. Oxford University Press, 1999. ISBN: 9780195135329. URL: https://books.google.pt/books?id=LT1RXREvkGIC (cit. on p. 55).

[46] L. Harrison et al. "Ranking Visualizations of Correlation Using Weber's Law". In: *Visualization and Computer Graphics, IEEE Transactions on* 20 (2014-12), pp. 1943–1952. DOI: 10.1109/TVCG.2014.2346979 (cit. on p. 55).

[47] J. Heer and B. Shneiderman. "Interactive Dynamics for Visual Analysis". In: *Communications of the ACM* 55 (2012-04), pp. 45–54. DOI: 10.1145/2133806.2133821 (cit. on p. 18).

[48] R. Herbrich, T. Graepel, and K. Obermayer. "Support vector learning for ordinal regression". In: *1999 Ninth International Conference on Artificial Neural Networks ICANN 99. (Conf. Publ. No. 470)*. Vol. 1. 1999, 97–102 vol.1. DOI: 10.1049/cp:19991091 (cit. on p. 72).

[49] K. Z. Hu et al. "VizML: A Machine Learning Approach to Visualization Recommendation". In: *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (2018) (cit. on pp. 24, 25, 31).

[50] A. Q. Jiang et al. *Mixtral of Experts*. 2024. arXiv: 2401.04088 [cs.LG] (cit. on p. 84).

[51] F. Jouault and J. Bézivin. "KM3: A DSL for metamodel specification". In: vol. 4037. 2006-05, pp. 171–185. ISBN: 978-3-540-34893-1. DOI: 10.1007/11768869_14 (cit. on p. 34).

[52] P. Kaur and M. Owonibi. "A Review on Visualization Recommendation Strategies". In: 2017-02. DOI: 10.5220/0006175002660273 (cit. on p. 24).

[53] M. Kay and J. Heer. "Beyond Weber's Law: A Second Look at Ranking Visualizations of Correlation". In: *IEEE Transactions on Visualization and Computer Graphics* 22.1 (2016), pp. 469–478. DOI: 10.1109/TVCG.2015.2467671 (cit. on p. 55).

[54] D. Keim et al. "Challenges in Visual Data Analysis". In: *Tenth International Conference on Information Visualisation (IV'06)*. 2006, pp. 9–16. DOI: 10.1109/IV.2006.31 (cit. on p. 21).

[55] D. Keim et al. "Visual Analytics: Scope and Challenges". In: *First publ. in: Lecture notes in computer science, No. 4404 (2008), pp. 76-90* 1 (2008-04). DOI: 10.1007/978-3-540-71080-6_6 (cit. on p. 22).

[56] M. Kim et al. "Data Scientists in Software Teams: State of the Art and Challenges". In: *IEEE Transactions on Software Engineering* 44.11 (2018), pp. 1024–1038. DOI: 10.1109/TSE.2017.2754374 (cit. on p. 108).

[57] G. Kindlmann, C. Scheidegger, and C. Scheidegger. "An Algebraic Process for Visualization Design". In: *IEEE Transactions on Visualization and Computer Graphics* (2014). DOI: 10.1109/tvcg.2014.2346325 (cit. on p. 12).

[58] A. Kobsa. "An empirical comparison of three commercial information visualization systems". In: *IEEE Symposium on Information Visualization, 2001. INFOVIS 2001.* 2001, pp. 123–130. DOI: 10.1109/INFVIS.2001.963289 (cit. on p. 75).

[59] T. Kosar et al. "Comparing General-Purpose and Domain-Specific Languages: An Empirical Study". In: *Computer Science and Information Systems* 438 (2010-05). DOI: 10.2298/CSIS1002247K (cit. on p. 34).

[60] T. Kühne. "Matters of (Meta-) Modeling". In: *Software & Systems Modeling* 5 (2006-12), pp. 369–385. DOI: 10.1007/s10270-006-0017-9 (cit. on p. 31).

[61] B. Laha et al. "A classification of user tasks in visual analysis of volume data". In: *2015 IEEE Scientific Visualization Conference (SciVis)*. 2015, pp. 1–8. DOI: 10.1109/SciVis.2015.7429485 (cit. on pp. 42, 46, 47, 49).

[62] E. Lees. *Altair-Stack*. 2024-03-20. 2020. URL: https://eitanlees.github.io/altair-stack/ (cit. on p. 67).

[63] A. Leon and O. Pastor. "Towards a Shared, Conceptual Model-Based Understanding of Proteins and Their Interactions". In: *IEEE Access* 9 (2021), pp. 73608–73623. DOI: 10.1109/ACCESS.2021.3080040 (cit. on p. 36).

[64] H. Li et al. "KG4Vis: A Knowledge Graph-Based Approach for Visualization Recommendation". In: *IEEE Transactions on Visualization and Computer Graphics* 28.1 (2022-01), pp. 195–205. ISSN: 2160-9306. DOI: 10.1109/tvcg.2021.3114863. URL: http://dx.doi.org/10.1109/TVCG.2021.3114863 (cit. on pp. 26, 27).

[65] J. Li, J.-B. Martens, and J. J. van Wijk. "A Model of Symbol Size Discrimination in Scatterplots". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '10. Atlanta, Georgia, USA: Association for Computing Machinery, 2010, pp. 2553–2562. ISBN: 9781605589299. DOI: 10.1145/1753326.1753714. URL: https://doi.org/10.1145/1753326.1753714 (cit. on p. 55).

[66] T. Li, X. Wei, and Y. Wang. "A Requirements-Driven Framework for Automatic Data Visualization". In: *Enterprise, Business-Process and Information Systems Modeling*. Ed. by H. van der Aa et al. Cham: Springer Nature Switzerland, 2023, pp. 297–311. ISBN: 978-3-031-34241-7 (cit. on pp. 44–47, 49).

[67] J. M. Lourenço. *The NOVAthesis LaTeX Template User's Manual*. NOVA University Lisbon. 2021. URL: https://github.com/joaomlourenco/novathesis/raw/main/template.pdf (cit. on p. i).

[68] R. Lukyanenko et al. "Using Conceptual Modeling to Support Machine Learning". In: 2019-05, pp. 170–181. ISBN: 978-3-030-21296-4. DOI: 10.1007/978-3-030-21297-1_15 (cit. on p. 35).

[69] Y. Luo et al. "DeepEye: Towards Automatic Data Visualization". In: 2018-04, pp. 101–112. DOI: 10.1109/ICDE.2018.00019 (cit. on pp. 24, 28).

[70] J. Mackinlay. "Automating the design of graphical presentations of relational information". In: 5.2 (1986-04), pp. 110–141. ISSN: 0730-0301. DOI: 10.1145/22949.22950. URL: https://doi.org/10.1145/22949.22950 (cit. on pp. 14, 72).

[71] J. Mackinlay, P. Hanrahan, and C. Stolte. "Show Me: Automatic Presentation for Visual Analysis". In: *IEEE Transactions on Visualization and Computer Graphics* 13.6 (2007), pp. 1137–1144. DOI: 10.1109/TVCG.2007.70594 (cit. on p. 23).

[72] C. M. Martos, A. de Antonio Jiménez, and X. F. Grau. "Modeling the visualization and exploration of document collections with user and purpose-based adaptation". PhD thesis. Universidad Politécnica de Madrid, 2016 (cit. on p. 36).

[73] H. Mayr and B. Thalheim. "The triptych of conceptual modeling". In: *Software and Systems Modeling* 20 (2021-02). DOI: 10.1007/s10270-020-00836-z (cit. on pp. 34, 35).

[74] Microsoft. *Microsoft Copilot: Your everyday AI companion*. https://www.microsoft.com/en-us/microsoft-copilot. Accessed: 2024-02-17. 2024 (cit. on p. 84).

[75] N. Mihindukulasooriya et al. *Text2KGBench: A Benchmark for Ontology-Driven Knowledge Graph Generation from Text*. 2023. arXiv: 2308.02357 [cs.CL] (cit. on p. 108).

[76] C. Moral et al. "A proposed UML-based common model for information visualization systems". In: *Multimedia Tools and Applications* 80 (2021-03), pp. 1–39. DOI: 10.1007/s11042-020-10306-9 (cit. on p. 37).

[77] D. Moritz et al. "Formalizing Visualization Design Knowledge as Constraints: Actionable and Extensible Models in Draco". In: *IEEE Trans. Visualization & Comp. Graphics (Proc. InfoVis)* (2019). URL: http://idl.cs.washington.edu/papers/draco (cit. on p. 72).

[78] E. MORSE, M. LEWIS, and K. OLSEN. "Evaluating visualizations: using a taxonomic guide". In: *International Journal of Human-Computer Studies* 53.5 (2000), pp. 637–662. ISSN: 1071-5819. DOI: https://doi.org/10.1006/ijhc.2000.0412. URL: https://www.sciencedirect.com/science/article/pii/S1071581900904129 (cit. on p. 41).

[79] A. Mosca, A. Ottley, and R. Chang. "Does Interaction Improve Bayesian Reasoning with Visualization?" In: *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. CHI '21. Yokohama, Japan: Association for Computing Machinery, 2021. ISBN: 9781450380966. DOI: 10.1145/3411764.3445176. URL: https://doi.org/10.1145/3411764.3445176 (cit. on p. 11).

[80] T. Munzner. *Visualization analysis and design*. A.K Peters/CRC Press, 2014. ISBN: 978-0-429-08890-2. DOI: https://doi.org/10.1201/b17511 (cit. on pp. 3, 12, 13, 15–17, 20, 39, 40, 45–47, 49, 54, 57).

[81] J. Mylopoulos. "Conceptual Modelling and Telos1". In: (1992) (cit. on pp. 34, 35).

[82] M. M. Najafabadi et al. "Deep learning applications and challenges in big data analytics". In: *Journal of Big Data* (2015). DOI: 10.1186/s40537-014-0007-7 (cit. on p. 7).

[83] Object Management Group. *OMG Object Constraint Language (OCL)*. URL: https://www.omg.org/spec/OCL/2.4 (visited on 2024-03-29) (cit. on p. 60).

[84] *Object Management Group Model Driven Architecture (MDA) MDA Guide rev. 2.0*. URL: https://www.omg.org/cgi-bin/doc?ormsc/14-06-01 (visited on 2023-05-23) (cit. on p. 32).

[85] A. Olivé. *Conceptual Modeling of Information Systems*. 2007-01. ISBN: 978-3-540-39389-4. DOI: 10.1007/978-3-540-39390-0 (cit. on p. 35).

[86] *OMG® Unified Modeling Language® (OMG UML®) Version 2.5.1*. URL: https://www.omg.org/spec/UML/2.5.1/PDF (visited on 2023-05-25) (cit. on p. 34).

[87] B. Ondov et al. "Face to Face: Evaluating Visual Comparison". In: *IEEE Transactions on Visualization and Computer Graphics* 25.1 (2019), pp. 861–871. DOI: 10.1109/TVCG.2018.2864884 (cit. on p. 55).

[88] OpenAI. *ChatGPT*. https://openai.com/. Version 4. Accessed: 2024-02-17. 2024 (cit. on pp. 79, 84).

[89] OpenAI et al. *GPT-4 Technical Report*. 2023. arXiv: 2303.08774 [cs.CL] (cit. on p. 84).

[90] J. Poole. "Model-Driven Architecture: Vision, Standards And Emerging Technologies". In: (2001-01) (cit. on p. 33).

[91] X. Qin et al. "Making data visualization more efficient and effective: a survey". In: *The VLDB Journal* 29 (2020-01). DOI: 10.1007/s00778-019-00588-3 (cit. on pp. 1, 12).

[92] G. J. Quadri and P. Rosen. "A Survey of Perception-Based Visualization Studies by Task". In: *CoRR* abs/2107.07477 (2021). arXiv: 2107.07477. URL: https://arxiv.org/abs/2107.07477 (cit. on pp. 39, 42, 45–47, 49, 55, 56).

[93] S. Rendgen et al. *Information Graphics*. Taschen, 2012. ISBN: 9783836528795. URL: https://books.google.pt/books?id=cFeMZwEACAAJ (cit. on p. 44).

[94] J. Reyes Román et al. "Applying Conceptual Modeling to Better Understand the Human Genome". In: 2016-11. DOI: 10.1007/978-3-319-46397-1_31 (cit. on p. 34).

[95] J. Reyes Román et al. "Including haplotypes treatment in a Genomic Information Systems Management". In: 2016-04 (cit. on p. 35).

[96] J. Roberts. "Display models: ways to classify visual representations". In: *CIDAC, Volume 2 Issue 4 November 2000, pp. 242-250* (2000-01) (cit. on p. 9).

[97] P. Russom. "Big Data Analytics. TDWI Best Practices Report". In: (2011), pp. 1–34 (cit. on p. 8).

[98] D. Sacha et al. "Knowledge Generation Model for Visual Analytics". In: *Visualization and Computer Graphics, IEEE Transactions on* 20 (2014-12), pp. 1604–1613. DOI: 10.1109/TVCG.2014.2346481 (cit. on pp. 11, 21).

[99] B. Saket, A. Endert, and Ç. Demiralp. "Task-Based Effectiveness of Basic Visualizations". In: *IEEE Transactions on Visualization and Computer Graphics* 25.7 (2019), pp. 2505–2512. DOI: 10.1109/TVCG.2018.2829750 (cit. on p. 55).

[100] B. Saket et al. "Evaluating Interactive Graphical Encodings for Data Visualization". In: *IEEE Transactions on Visualization and Computer Graphics* 24.3 (2018), pp. 1316–1330. DOI: 10.1109/TVCG.2017.2680452 (cit. on p. 55).

[101] S. Santos and K. Brodlie. "Gaining understanding of multivariate and multidimensional data through visualization". In: *Computers & Graphics* 28 (2004-06), pp. 311–325. DOI: 10.1016/j.cag.2004.03.013 (cit. on p. 11).

[102] A. Sarikaya and M. Gleicher. "Scatterplots: Tasks, Data, and Designs". In: *IEEE Transactions on Visualization and Computer Graphics* 24.1 (2018), pp. 402–412. DOI: 10.1109/TVCG.2017.2744184 (cit. on p. 38).

[103] A. Sarikaya and M. Gleicher. "Scatterplots: Tasks, Data, and Designs". In: *IEEE Transactions on Visualization and Computer Graphics* 24.1 (2018), pp. 402–412. DOI: 10.1109/TVCG.2017.2744184 (cit. on p. 55).

[104] A. Satyanarayan et al. "Vega-Lite: A Grammar of Interactive Graphics". In: *IEEE Trans. Visualization & Comp. Graphics (Proc. InfoVis)* (2017). URL: http://idl.cs.washington.edu/papers/vega-lite (cit. on pp. 26, 27, 66, 67, 73).

[105] D. C. Schmidt et al. "Model-driven engineering". In: *Computer-IEEE Computer Society-* 39.2 (2006), p. 25 (cit. on p. 32).

[106] H.-J. Schulz et al. "A Design Space of Visualization Tasks". In: *IEEE Transactions on Visualization and Computer Graphics* 19.12 (2013), pp. 2366–2375. DOI: 10.1109/TVCG.2013.120 (cit. on pp. 1, 10, 17).

[107] S. Sendall and W. Kozaczynski. "Model transformation: the heart and soul of model-driven software development". In: *IEEE Software* 20.5 (2003), pp. 42–45. DOI: 10.1109/MS.2003.1231150 (cit. on p. 33).

[108] B. Shneiderman. "The eyes have it: a task by data type taxonomy for information visualizations". In: *Proceedings 1996 IEEE Symposium on Visual Languages*. 1996, pp. 336–343. DOI: 10.1109/VL.1996.545307 (cit. on pp. 38, 41, 109).

[109] R. R. Springmeyer, M. M. Blattner, and N. L. Max. "A Characterization of the Scientific Data Analysis Process". In: *Proceedings of the 3rd Conference on Visualization '92*. VIS '92. Boston, Massachusetts: IEEE Computer Society Press, 1992, pp. 235–242. ISBN: 0818628960 (cit. on p. 41).

[110] D. Streeb et al. "Why Visualize? Untangling a Large Network of Arguments". In: *IEEE Transactions on Visualization & Computer Graphics* 27.03 (2021), pp. 2220–2236. ISSN: 1941-0506. DOI: 10.1109/TVCG.2019.2940026 (cit. on p. 12).

[111] C. Sun et al. "How to fine-tune bert for text classification?" In: *Chinese Computational Linguistics: 18th China National Conference, CCL 2019, Kunming, China, October 18–20, 2019, Proceedings 18*. Springer. 2019, pp. 194–206 (cit. on p. 110).

[112] I. Sutskever, O. Vinyals, and Q. V. Le. *Sequence to Sequence Learning with Neural Networks*. 2014. arXiv: 1409.3215 [cs.CL] (cit. on p. 26).

[113] D. A. Szafir et al. "Four types of ensemble coding in data visualizations". In: *Journal of Vision* 16.5 (2016-03), pp. 11–11. ISSN: 1534-7362. DOI: 10.1167/16.5.11. URL: https://doi.org/10.1167/16.5.11 (cit. on p. 56).

[114] Tableau Software. *Superstore Dataset*. https://www.tableau.com/solutions/gallery/superstore. Accessed: yyyy-mm-dd (cit. on pp. 95, 102).

[115] G. Team et al. *Gemini: A Family of Highly Capable Multimodal Models*. 2023. arXiv: 2312.11805 [cs.CL] (cit. on p. 84).

[116] E. R. A. Valiati, M. S. Pimenta, and C. M. D. S. Freitas. "A Taxonomy of Tasks for Guiding the Evaluation of Multidimensional Visualizations". In: *Proceedings of the 2006 AVI Workshop on BEyond Time and Errors: Novel Evaluation Methods for Information Visualization*. BELIV '06. Venice, Italy: Association for Computing Machinery, 2006, pp. 1–6. ISBN: 1595935622. DOI: 10.1145/1168149.1168169. URL: https://doi.org/10.1145/1168149.1168169 (cit. on pp. 39, 41, 45–47, 49, 109).

[117] J. VanderPlas et al. "Altair: Interactive Statistical Visualizations for Python". In: *Journal of Open Source Software* 3.32 (2018), p. 1057. DOI: 10.21105/joss.01057. URL: https://doi.org/10.21105/joss.01057 (cit. on pp. 27, 66, 67).

[118] M. Vartak et al. "Towards Visualization Recommendation Systems". In: *SIGMOD Rec.* 45.4 (2017-05), pp. 34–39. ISSN: 0163-5808. DOI: 10.1145/3092931.3092937. URL: https://doi.org/10.1145/3092931.3092937 (cit. on p. 24).

[119] A. Vaswani et al. "Attention is All you Need". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf (cit. on p. 53).

[120] A. Vázquez-Ingelmo et al. "Representing Data Visualization Goals and Tasks Through Meta-Modeling to Tailor Information Dashboards". In: *Applied Sciences* 10 (2020-03). DOI: 10.3390/app10072306 (cit. on p. 1).

[121] M. Vigni, C. Durante, and M. Cocchi. "Exploratory Data Analysis". In: *Data Handling in Science and Technology* 28 (2013-12), pp. 55–126. DOI: 10.1016/B978-0-444-59528-7.00003-X (cit. on p. 8).

[122] B. Wang et al. *Grammar Prompting for Domain-Specific Language Generation with Large Language Models*. 2023. arXiv: 2305.19234 [cs.CL] (cit. on p. 82).

[123] Y. Wang et al. "DataShot: Automatic Generation of Fact Sheets from Tabular Data". In: *IEEE Transactions on Visualization and Computer Graphics* (2019-11). URL: https://www.microsoft.com/en-us/research/publication/datashot-automatic-generation-of-fact-sheets-from-tabular-data/ (cit. on p. 23).

[124] M. Ward, G. Grinstein, and D. Keim. *Interactive Data Visualization: Foundations, Techniques, and Applications, Second Edition*. 360 Degree Business. CRC Press, 2015. ISBN: 9781482257380. URL: https://books.google.pt/books?id=XHZ3CAAAQBAJ (cit. on p. 17).

[125] C. Ware. "Information Visualization: Perception for Design: Second Edition". In: 2004-04 (cit. on p. 16).

[126] U. of Washington Data Lab. *Draco-Learn: Learning Constraints for the Draco Framework*. https://github.com/uwdata/draco-learn. Accessed: 2024-01-07. 2018 (cit. on p. 72).

[127] S. Wehrend and C. H. Lewis. "A problem-oriented classification of visualization techniques". In: *Proceedings of the First IEEE Conference on Visualization: Visualization '90* (1990), pp. 139–143 (cit. on pp. 17, 39, 44–47, 49, 52).

[128] J. van Wijk. "The value of visualization". In: *VIS 05. IEEE Visualization, 2005.* 2005, pp. 79–86. DOI: 10.1109/VISUAL.2005.1532781 (cit. on p. 11).

[129] L. Wilkinson. *The Grammar of Graphics (Statistics and Computing)*. Berlin, Heidelberg: Springer-Verlag, 2005. ISBN: 0387245448 (cit. on pp. 23, 67, 68).

[130] K. Wongsuphasawat et al. "Voyager 2: Augmenting Visual Analysis with Partial View Specifications". In: *ACM Human Factors in Computing Systems (CHI)*. 2017. URL: http://idl.cs.washington.edu/papers/voyager2 (cit. on p. 29).

[131] K. Wongsuphasawat et al. "Voyager: Exploratory Analysis via Faceted Browsing of Visualization Recommendations". In: *IEEE Transactions on Visualization and Computer Graphics* 22.1 (2016), pp. 649–658. DOI: 10.1109/TVCG.2015.2467191 (cit. on pp. 23, 29).

[132] J. Yang et al. *Draco 2: An Extensible Platform to Model Visualization Design*. IEEE, 2023. eprint: arXiv:2308.14247 (cit. on pp. 27, 66).

[133] N. Yau. *Visualize this: The FlowingData Guide to Design, Visualization, and Statistics*. Wiley Pub., 2011. ISBN: 9781118722213. URL: https://books.google.pt/books?id=otpRtAEACAAJ (cit. on pp. 39, 43–47, 49).

[134] Z. Zeng and L. Battle. "A Review and Collation of Graphical Perception Knowledge for Visualization Recommendation". In: *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. CHI '23. <conf-loc>, <city>Hamburg</city>, <country>Germany</country>, </conf-loc>: Association for Computing Machinery, 2023. ISBN: 9781450394215. DOI: 10.1145/3544548.3581349. URL: https://doi.org/10.1145/3544548.3581349 (cit. on p. 72).

[135] Z. Zeng et al. "An Evaluation-Focused Framework for Visualization Recommendation Algorithms". In: *IEEE Transactions on Visualization and Computer Graphics* 28.1 (2022), pp. 346–356. DOI: 10.1109/TVCG.2021.3114814 (cit. on p. 72).

[136] M. Zhou et al. "Table2Charts: Recommending Charts by Learning Shared Table Representations". In: *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. KDD '21. ACM, 2021-08. DOI: 10.1145/3447548.3467279. URL: http://dx.doi.org/10.1145/3447548.3467279 (cit. on pp. 30, 31).

[137] M. X. Zhou and S. K. Feiner. "Visual Task Characterization for Automated Visual Discourse Synthesis". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '98. Los Angeles, California, USA: ACM Press/Addison-Wesley Publishing Co., 1998, pp. 392–399. ISBN: 0201309874. DOI: 10.1145/274644.274698. URL: https://doi.org/10.1145/274644.274698 (cit. on pp. 17, 41).

[138] S. Zhu et al. "A survey on automatic infographics and visualization recommendations". In: *Visual Informatics* 4.3 (2020), pp. 24–40. ISSN: 2468-502X. DOI: https://doi.org/10.1016/j.visinf.2020.07.002. URL: https://www.sciencedirect.com/science/article/pii/S2468502X20300292 (cit. on pp. 22, 24, 72).

I

LLM Testing Prompts

Testing on phase 2 was executed following the same set of steps for every model. Each LLM received a general text explaining the context regarding our testing and our taxonomy (Figure I.1 and I.2), and then received a couple of learning examples for each analytical task available (Figure I.3). For each distinct domain, the LLM received a small dataset sample for added context. Finally, there will be two examples of a prompt with a domain question and the respective answer provided by the LLM, in Figure I.4 and I.5. In this annex, all figures are relative to conversation logs with GPT-4. Important to note that due to a late change in the taxonomy names, these logs still have analytical tasks such as "Relationship", "Pattern", and "Grouping".

Anonymous

The context of this work is: that there's a necessity to analyze and translate questions relative to specific domains (domain questions), into a set of generic analytical tasks. And said tasks are capable of being used later on to create visualizations. A set of tasks can have multiple complementary (C) and visualization (V) tasks.
Currently, I need you to help me translate domain questions into a single or group of analytical tasks. Take into consideration these so-denominated "Analytical Tasks". Some Analytical Tasks have Subtasks and others do not. The tasks available are the following:

Identify - Defines any action related to the necessity of finding and identifying insights on the dataset. It is capable of identifying patterns, relationships, extremes, outliers, and clusters of data.
Task 1 of Identify: Relationship (V) - Allows the identification of relationships and interactions between data elements and, consequently finds insights about data.
Task 2 of Identify: Pattern (V) - Identifies patterns, these patterns are found as trends that can be found along certain intervals in the data.Task 3 of Identify: Find Extreme (V) - Focuses on finding values that can be considered extremes within a specific interval of data.
Task 4 of Identify: Find Anomalies (V) - Focuses on finding values that can be considered outliers within a specific interval of the domain.
Task 5 of Identify: Find Clusters (V) - Focuses on finding clusters of data within the dataset.

Figure I.1: First part of two, of a conversation log, with the GPT-4 model, related to analytical task learning.

Task 6: Comparison (V) - Compares values of data elements or groups of data elements, between each other.

Task 7: Determine Distribution (V) - It presents an analytical distribution of the values for a certain attribute.

Organize - Organize data by following user-defined heuristics.
Task 8 of Organize: Sort (C) - Arrange items in a specific order based on criteria defined by the user.
Task 9 of Organize: Rank (C) - Assigns a numerical position to each element and orders them based on that same numerical position.

Derive - Sometimes data doesn't provide enough necessary information, therefore there is a need to derive new attributes, new elements, and new datasets from already existing data in order to perform a better analysis.
Task 10 of Derive: Grouping (C) - Groups elements together. These groups are created in a deterministic way in which the user decides the criteria of creation.
Task 11 of Derive: Transform (C) - Function that allows for a transformation to occur into all the elements of a certain group or dataset.
Task 12 of Derive: Aggregate (C) - Function that processes data in a way that expresses a summarization of elements in a statistical way, such as a median, sum, count, etc...
Task 13 of Derive: Categorize (C) - Creates a label that categorizes and characterizes elements. These labels are usually created in concordance with the domain concepts.
Task 14 of Derive: Compute Attribute (C) - Allows the creation of a new attribute based on already existing data.

Figure I.2: Second part of two, of a conversation log, with the GPT-4 model, related to analytical task learning.

Anonymous

Take into consideration the following examples of each of the tasks:
- Relationship: "Is there a connection between time spent training and exercise performance ?".
- Pattern: "Is there an existing pattern between sales number and the month of the year ?"
- Find Extreme: "What are the cars with the highest mileage, in the dealership ?"
- Find Anomalies: "Are there existing exceptions to the median summer temperature over the years ?"
- Find Clusters: "Do groups of ingredients share similar fiber quantities ?"
- Compare: "How do the apple prices change in different supermarkets ?"
- Determine Distribution: "What is the age distribution of the shop clients ?"
- Rank: "What is the ordered classification of the championship ?"
- Sort: "Show the clients that have the highest packet weight per delivery ?"
- Transform: "How would a new tax rate influence workers salaries ?"
- Aggregate: "What is the gross expenses of all factories combined ?"
- Categorize: "What is the category (class) of each manufactured car ?"
- Compute Attribute: "Create the profit ratio of each order ?"

Figure I.3: Providing a set of examples for the model to take into consideration when learning the necessary context.

I will now give you prompt-by-prompt multiple domain questions to which I would like you to provide me the necessary analytical task(s) to provide an adequate answer to said domain question. Starting with: "Which cities are the furthest west and east cities ?"

ChatGPT

To answer the domain question "Which cities are the furthest west and east cities?", you will need to engage with the following analytical task(s):

- **Identify: Find Extreme (V)** - This task is directly applicable because it focuses on finding values that can be considered extremes within a specific interval of data. In this case, the "interval" refers to the longitudinal values (represented by the `lng` attribute in your dataset) of cities around the globe. The task involves identifying the cities with the minimum (furthest west) and maximum (furthest east) longitude values.

This task will help in identifying the geographical extremes among the cities listed in your dataset based on their longitudinal positioning.

Figure I.4: Providing a domain question, of a spatial domain, to a model and its respective answer.

What if we are looking into domain questions that are a bit more intricate and interpretative, such as: "What are the individual REF and ALT alleles of each chromosome for each genotype ?"

ChatGPT

For a more intricate and interpretative question like "What are the individual REF and ALT alleles of each chromosome for each genotype?" the necessary analytical tasks would involve a combination of data organization and derivation processes to segregate and summarize the data appropriately:

1. **Derive: Grouping (C)** - Group data first by chromosome ("CHROM") and then within each chromosome group by genotype ("GENOTYPE"). This hierarchical grouping is essential to organize the data for detailed analysis.
2. **Derive: Aggregate (C)** - Within each group defined by chromosome and genotype, aggregate data to compile a list or summary of the REF (reference allele) and ALT (alternative allele) combinations. This could involve creating a concatenated or structured representation of each unique REF and ALT pair found within the group.
3. **Organize: Sort (C)** or **Organize: Rank (C)** - Optionally, to enhance interpretability, you might sort or rank the groups either by chromosome position ("POS") for a sequential overview or by the frequency of genotype occurrences to highlight the most common REF and ALT combinations within each chromosome and genotype category.

These tasks will enable the detailed breakdown of REF and ALT alleles across different chromosomes and genotypes, providing a comprehensive overview that supports intricate analysis and interpretation of genetic variation patterns within the dataset.

Figure I.5: Providing a domain question, of a genome domain, to a model and its respective answer.

II

GitHub Repository

This annex section will provide a hyperlink to the current Github repository that keeps important data concerning the work done throughout this dissertation.

In this **repository** the following content will be accessible:

- **Recommendation System** - The folder has the modified instance of Draco, its testing files, and the parser allowing for automatic visualization generation.
- **Meta-Model** - It has a designated folder with the files for the model, model constraints, and images of the individual components.
- **Data Repository** - Group of files and folders that build up the data repository to bridge the abstractions from the meta-model into the recommendation system. The data repository currently has six working domains: Business, car manufacturing, socioeconomic, temporal, spatial, and genomics.
- **Testing** - The findings relative to all the LLM testing, both phases 1 and 2, and the ground truth dataset created from these findings can be used to further analyze our results. All the tableau workbooks used to generate our graphics will also be present. Also, all the visualizations in the visualization generation testing section can be found inside this folder.
- **Draco Training Dataset** - Contains the file that organizes the dataset and the jupyter notebooks corresponding to the different domain questions, regarding each trained analytical task.

III

Visualization Testing Results

This annex makes available all the remaining visualizations that resulted from the Visualization Generation Section 6.2, but were not exhibited/featured. It will be organized by domain and instance order, similar to how it is mentioned in the section mentioned earlier.

III.1 Genome Domain

The domain was the first one to be tested and the only one featuring three instances. For the **first instance** the visualizations can be found in Figure III.1, **second instance** has its visualizations present in Figure III.2, and lastly the **third instance** has its remaining visualization in Figure III.3.

III.2 Business Domain

This domain was presented in the testing chapter as the second example and it presents two instances. The remaining visualizations from **first instance** can be found in Figure III.4 and the **second instance** ones can be found in Figure III.5.

III.3 Spatial Domain

Appears as the last tested domain in the testing chapter, represented by two instances. The visualizations that were not evaluated from the **first instance** are present in Figure III.6 and in Figure III.7 for the **second instance**.

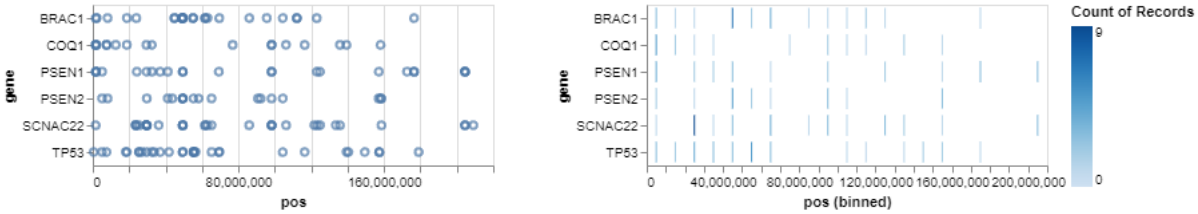

Figure III.1: Remaining visualizations from the first instance of the genome domain, in testing.

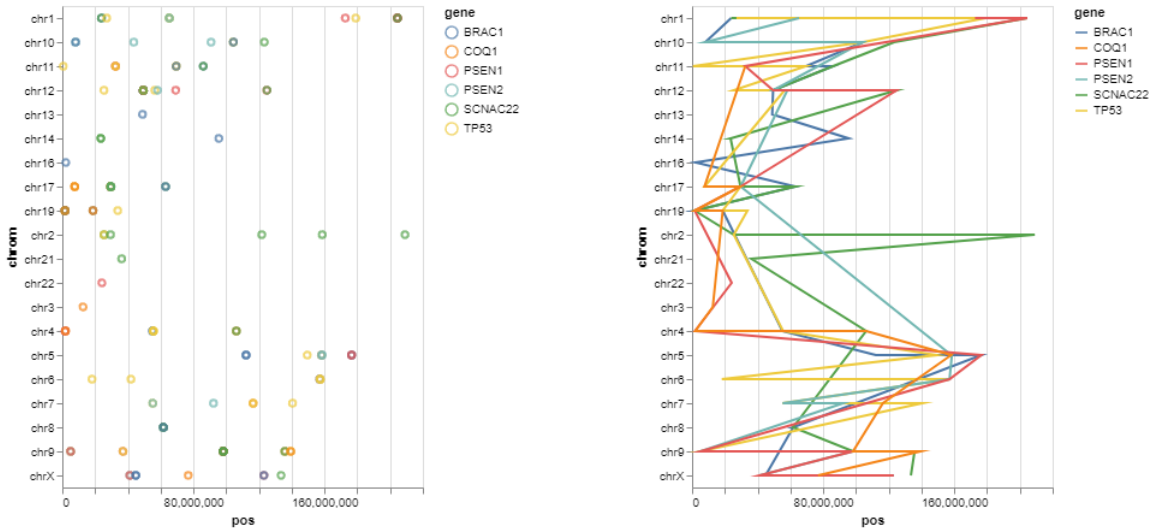

Figure III.2: Remaining visualizations from the second instance of the genome domain, in testing.

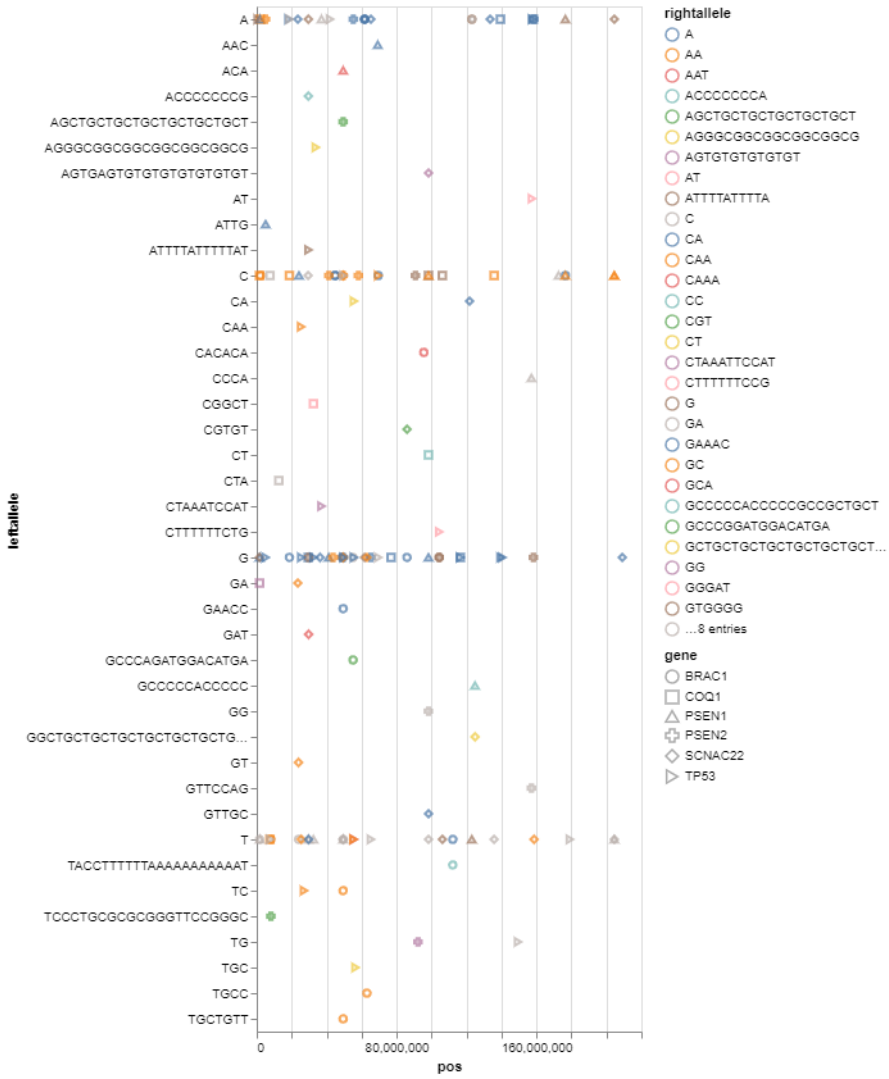

Figure III.3: Remaining visualization from the third instance of the genome domain, in testing.

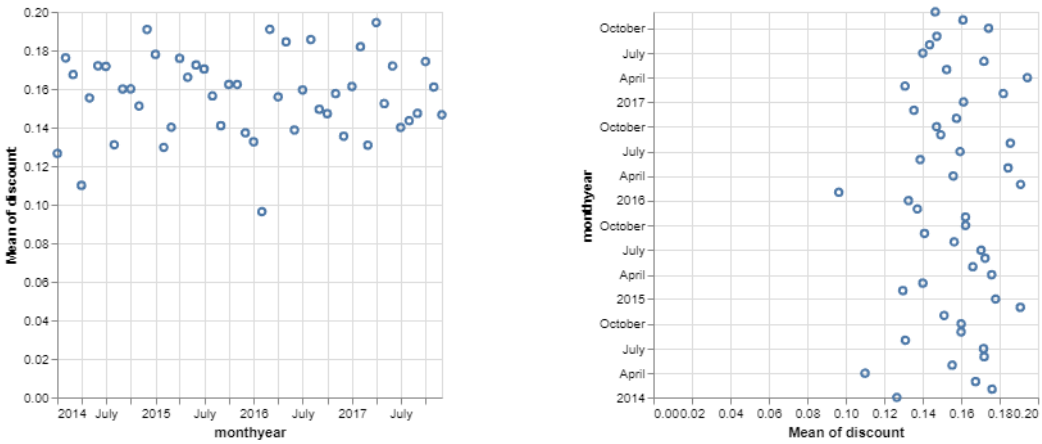

Figure III.4: Remaining visualizations from the first instance of the business domain, in testing.

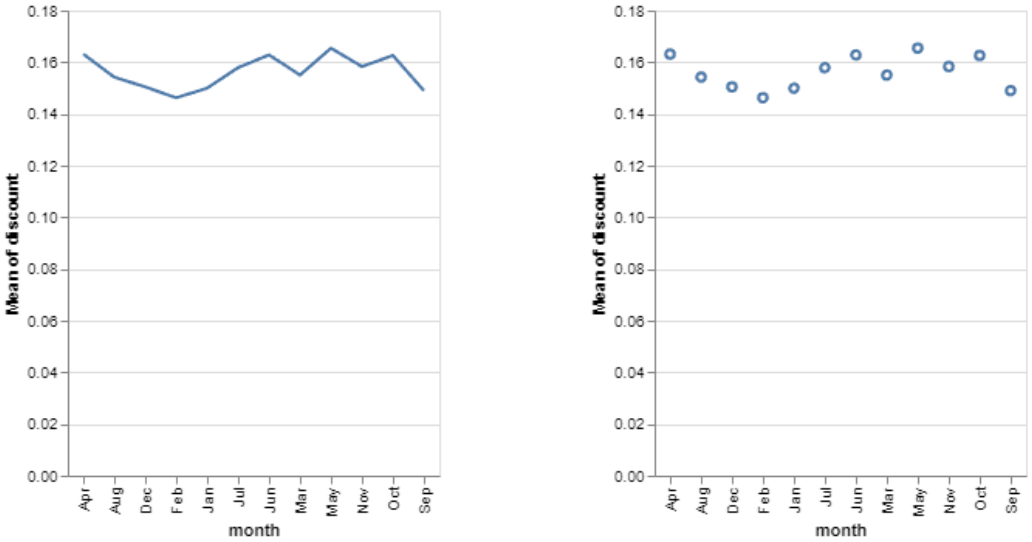

Figure III.5: Remaining visualizations from the second instance of the business domain, in testing.

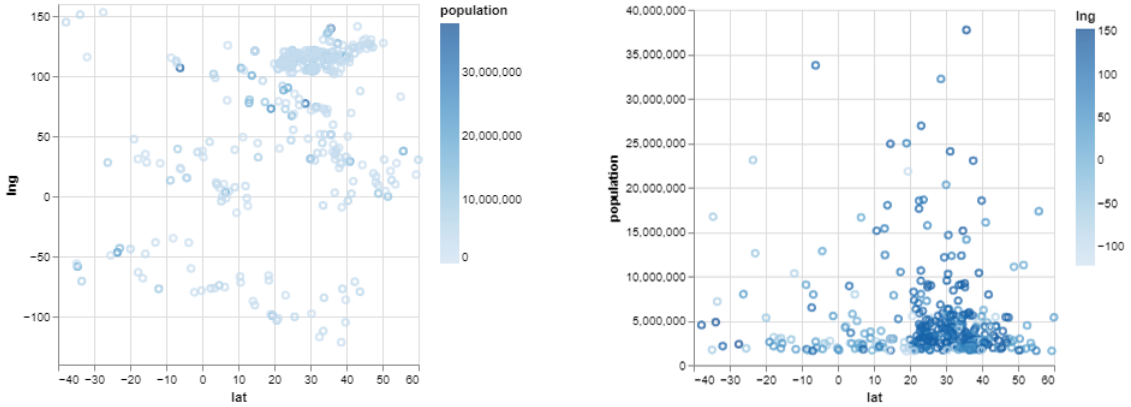

Figure III.6: Remaining visualizations from the first instance of the spatial domain, in testing.

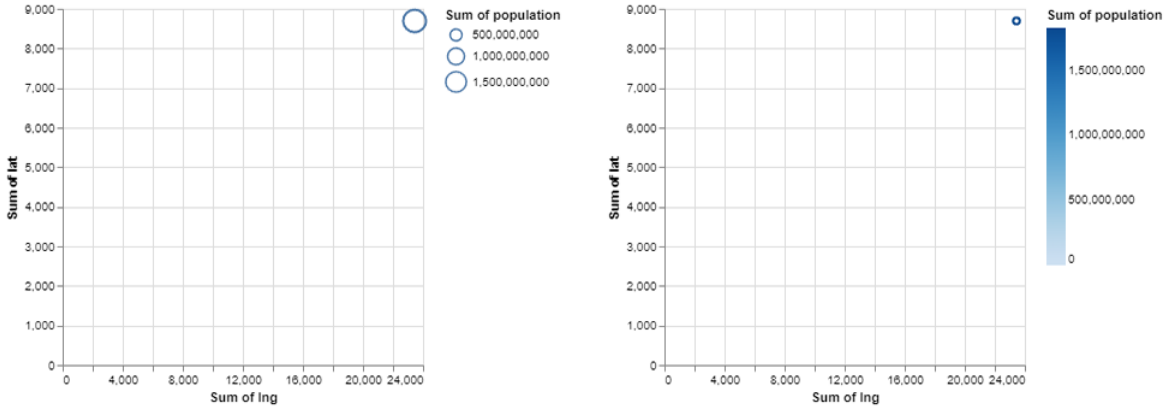

Figure III.7: Remaining visualizations from the second instance of the spatial domain, in testing.

IV

Complete Meta-Model

This annex chapter will present the meta-model and its respective constraints that were not shown as a whole or only as samples. It will feature both the two sets of constraints available in the model in Figures IV.1 and IV.2, and the meta-model, in a landscape page, in Figure IV.3

Constraints:

Context Relate
inv: self.Identify.AnalyticalTask.Chart.notEmpty()**implies**(self.Identify.AnalyticalTask.Chart.type = 'Scatterplot' or self.Identify.AnalyticalTask.Chart.type = 'LineChart' orself.Identify.AnalyticalTask.Chart.type = 'BarChart' or self.Identify.AnalyticalTask.Chart.type = 'HeatMap' orself.Identify.AnalyticalTask.Chart.type = 'HighlightTable' or self.Identify.AnalyticalTask.Chart.type = 'Map' orself.Identify.AnalyticalTask.Chart.type = 'SymbolMap')

ContextFindPatterns
inv: self.Identify.AnalyticalTask.Chart.notEmpty()**implies**(self.Identify.AnalyticalTask.Chart.type = 'Scatterplot' or self.Identify.AnalyticalTask.Chart.type = 'LineChart' orself.Identify.AnalyticalTask.Chart.type = 'HeatMap' or self.Identify.AnalyticalTask.Chart.type = '"HighlightTable' orself.Identify.AnalyticalTask.Chart.type = 'BarChart' or self.Identify.AnalyticalTask.Chart.type = 'AreaChart' orself.Identify.AnalyticalTask.Chart.type = 'Map' or self.Identify.AnalyticalTask.Chart.type = 'SymbolMap')

ContextFindExtremes
inv: self.Identify.AnalyticalTask.Chart.notEmpty()**implies**(self.Identify.AnalyticalTask.Chart.type = 'Scatterplot' or self.Identify.AnalyticalTask.Chart.type = 'LineChart' orself.Identify.AnalyticalTask.Chart.type = 'BarChart' or self.Identify.AnalyticalTask.Chart.type = 'AreaChart' orself.Identify.AnalyticalTask.Chart.type = 'Map' or self.Identify.AnalyticalTask.Chart.type = 'SymbolMap')

ContextFindAnomalies
inv: self.Identify.AnalyticalTask.Chart.notEmpty()**implies**(self.Identify.AnalyticalTask.Chart.type = 'Scatterplot' or self.Identify.AnalyticalTask.Chart.type = 'LineChart' orself.Identify.AnalyticalTask.Chart.type = 'Map' or self.Identify.AnalyticalTask.Chart.type = 'SymbolMap'

ContextFindClusters
inv: self.Identify.AnalyticalTask.Chart.notEmpty()**implies**(self.Identify.AnalyticalTask.Chart.type = 'Scatterplot')

ContextCompare:
inv: self.AnalyticalTask.Chart.notEmpty()**implies**(self.Identify.AnalyticalTask.Chart.type = 'Scatterplot' or self.Identify.AnalyticalTask.Chart.type = 'LineChart' orself.Identify.AnalyticalTask.Chart.type = 'BarChart' or self.Identify.AnalyticalTask.Chart.type = 'AreaChart' orself.Identify.AnalyticalTask.Chart.type = 'PackedBubbles' or self.Identify.AnalyticalTask.Chart.type = 'TextTable' orself.Identify.AnalyticalTask.Chart.type = 'HeatMap' or self.Identify.AnalyticalTask.Chart.type = 'PieChart' orself.Identify.AnalyticalTask.Chart.type = 'Map' or self.Identify.AnalyticalTask.Chart.type = 'SymbolMap' orself.Identify.AnalyticalTask.Chart.type = 'HighlightTable' or self.Identify.AnalyticalTask.Chart.type = 'CircleViews' orself.Identify.AnalyticalTask.Chart.type = 'BulletChart')

ContextDetermineDistribution:
inv: self.AnalyticalTask.Chart.notEmpty()**implies**(self.Identify.AnalyticalTask.Chart.type = 'Scatterplot' or self.Identify.AnalyticalTask.Chart.type = 'LineChart' orself.Identify.AnalyticalTask.Chart.type = 'HeatMap' or self.Identify.AnalyticalTask.Chart.type = 'BarChart' orself.Identify.AnalyticalTask.Chart.type = 'AreaChart' or self.Identify.AnalyticalTask.Chart.type = 'GanttChart' orself.Identify.AnalyticalTask.Chart.type = 'BulletChart')

ContextOrganize
inv: self.AnalyticalTask.Chart.isEmpty()

ContextDerive
inv: self.AnalyticalTask.Chart.isEmpty()

Figure IV.1: First set of constraints created, in the meta-model, to restrict the feasible visualizations for each analytical task.

Constraints:

Context Chart
inv: (self.nAxis = 0 and self.nHeader >= 1 and self.nMark >= 1)
implies
(self.type = 'TextTable' or self.type = 'HeatMap' or self.type = 'HighlightTable')

Context Chart
inv: (self.nAxis = 0 and self.nHeader = 0 and self.nMark >= 3)
implies
(self.type = 'PieChart' or self.type = 'TreeMap' or self.type = 'PackedBubbles')

Context Chart
inv: (self.nAxis = 2 and self.nHeader = 0 and self.Mark >= 2)
implies
(self.type = 'Scatterplot' or self.type = 'Histogram' or self.type = 'LineChart_especialization_Scatterplot'
or self.type = 'AreaChart_especialization_Scatterplot' or self.type = 'Map or self.type = 'SymbolMap')

Context Chart
inv: (self.nAxis= 1 and self.nHeader = 1 and self.nMark >=2)
implies
(self.type = 'BarChart' or self.type = 'GanttChart' or self.type = 'BulletChart' or self.type = 'CircleView' or
self.type = 'LineChart_especialization_BarChart' or self.type = 'AreaChat_especialization_BarChart')

Context Chart
inv: (self.type <> 'PieChart' or self.type <> 'ScatterPlot') **implies**self.ChartComponent.Mark.type =
'angle'

Figure IV.2: Second set of constraints created, in the meta-model, to arrange the minimum requirements for a specific type of chart.

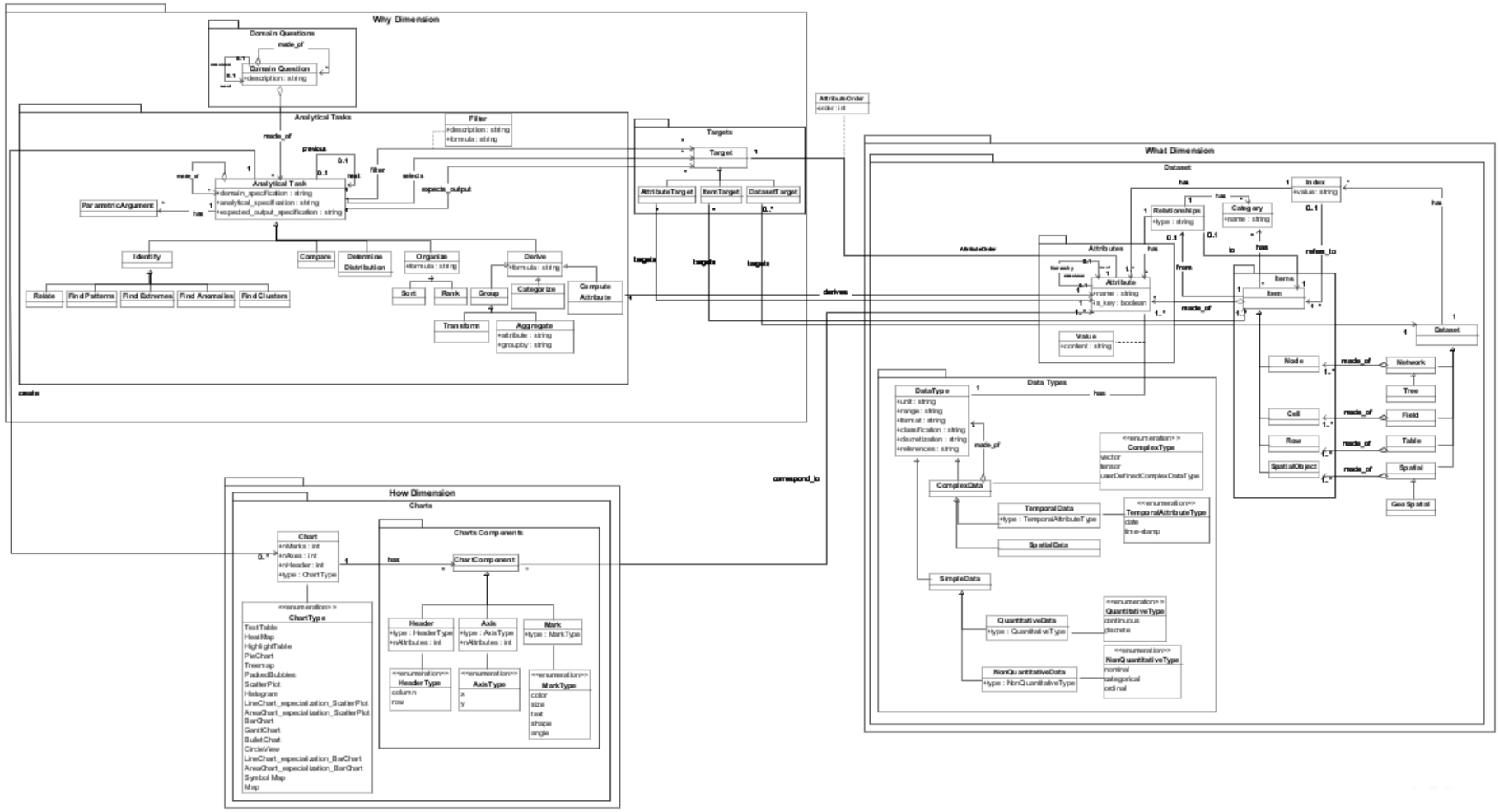

Figure IV.3: Complete display of our approach's meta-model.