The Pennsylvania State University

The Graduate School

College of Engineering

SEMANTIC STRUCTURING OF SCIENTIFIC INFORMATION IN SCHOLARLY DOCUMENTS

A Dissertation in

Computer Science and Engineering

by

Rabah Al-Zaidy

Submitted in Partial Fulfillment

of the Requirements

for the Degree of

Doctor of Philosophy

August 2017

The dissertation of Rabah Al-Zaidy was reviewed and approved* by the following:

*Signatures are on file in the Graduate School.

Abstract

The continuing growth of published scholarly content on the web ensures the availability of the most recent scientific findings to researchers. Scientific information extraction from these documents into a structured knowledge graph representation facilitates automated machine understanding of the documents. Knowledge graphs model information as entities that are semantically related. Thus, in order to restructure a scholarly document into such a representation, we must identify meaningful entities and relationships that capture the scientific facts within the articles as accurately as possible. In this thesis, we propose a suite of algorithms that are designed precisely for the task of semantic structuring of scientific content in scholarly documents. The thesis addresses two main areas of this problem. The first is concerned with algorithms capable of automatically understanding scientific charts in documents. These charts play an important role in scholarly documents as their content, most often than not, contains key facts that are not mentioned elsewhere in the document text. Scientific charts are an effective tool to visualize numerical data. They appear in a wide range of contexts, from experimental results in scientific papers to statistical analyses in business reports. The abundance of scientific charts in the web has made it inevitable for search engines to include them as indexed content. However, by relying solely on the meta data tags to understanding the charts the facts represented in the charts can not be fully available to information retrieval tools. Unfortunately, most applications, such as search indexing, use image meta-data to describe these charts rather than the information the graphic was initially designed to display.

Many studies exist to address the extraction of data from scientific diagrams in order to improve search results. Specifically, the problem of understanding digital charts found, specifically, in scholarly documents and inferring useful textual information from their graphical components is the focus of numerous studies.

In our approach to achieving this goal, we attempt to enhance the semantic labelling of scientific charts by using the original data values that these charts were designed to represent. In this work, we describe a framework to automatically read chart data, specifically bar charts, and provide the user with a textual summary of the chart. The chart reading process is fully automated using image processing and text recognition techniques combined with various heuristics derived from the graphical properties of bar charts to extract the original data values. The proposed framework follows a knowledge discovery approach that relies on a versatile graph representation of the chart. This representation is derived from analyzing a chart's original data values, from which useful features are extracted. The data features are in turn used to construct a semantic-graph. To illustrate the portability of the semantic graph structure we use a common natural language application, summary generation. To generate a summary, the semantic-graph of the chart is easily mapped to appropriately crafted protoforms, which are linguistic constructs based on fuzzy logic. We verify the effectiveness of our framework by conducting experiments on bar charts extracted from over $1,000$ PDF documents. Our preliminary results show that, under certain assumptions, 83% of the produced summaries provide plausible descriptions of the bar charts.

The second focus area of this thesis, is that of automatically understanding a document's scientific text itself. Specifically, we address the problem of constructing a knowledge graph from a set of scholarly documents that contains a large set of concepts found in scientific research content that can not be found in a general purpose knowledge base. Traditional information extraction approaches, that either require training samples or a preexisting knowledge base to assist in the extraction, can be challenging when applied to such repositories. Labeled training examples for such large scale are difficult to obtain for such datasets. Also, most available knowledge bases are built from web data and do not have sufficient coverage to include concepts found in scientific articles. In this thesis, we aim to construct a knowledge graph from scholarly documents while addressing both these issues. We propose a fully automatic, unsupervised system for scientific information extraction that does not build on an existing knowledge base and avoids manually-tagged training data. We describe and evaluate a constructed knowledge graph resulting from applying our approach to 10k documents.

Table of Contents

List of Figures **ix**

List of Tables **x**

Acknowledgments **xi**

Chapter 1
Introduction **1**
1.1 Motivation 5
1.1.1 Automatic Understanding of Bar Charts 6
1.1.2 Scientific Knowledge Base Construction 7
1.2 Objectives 10
1.3 Contribution 11
1.4 Thesis Organization 12

Chapter 2
Preliminaries **13**
2.1 Chart Data Extraction 13
2.1.1 Chart Figures in Scholarly Documents 14
2.1.2 Connected Component Analysis 15
2.2 Summary Generation 17
2.3 Knowledge Base Construction 18

Chapter 3
Bar Chart Data Extraction **21**
3.1 Rule-Based Data Extraction 21
3.1.1 Graphical Component Extraction 23
3.1.2 Text Component Extraction 24
3.1.2.1 Identification of Text Regions in a Chart 24
3.1.2.2 Extracting Text and Numerical Values 24

3.2 Machine Learning based Extraction 25
3.2.1 Graphic Component Extraction 26
3.2.2 Text Component Extraction 26
3.2.3 Component Role Classification 27
3.2.3.1 Graphic Component Classification 28
3.2.3.2 Text Region Classification 29
3.3 Chart Data Inference 31

Chapter 4
Semantic Graph Representation **34**
4.1 Data Analysis 34
4.2 Semantic Graph Construction 36
4.3 Automatic Summary Generation 37
4.3.1 Fuzzy Logic Constructs 37
4.3.2 Summary Generation Algorithm 39

Chapter 5
Scientific Knowledge Base Construction **40**
5.1 Problem Definition 40
5.2 System Overview 41
5.3 Hypernym Relation Extraction 42
5.3.1 Syntactic-based Extraction 43
5.3.2 Iterative Semantic Learning 43
5.4 Scientific Taxonomy Construction 45

Chapter 6
Evaluation of Chart Data Extraction **50**
6.1 Dataset 50
6.2 Rule-Based Extraction 51
6.3 Machine Learning-based Extraction 53
6.3.1 Graphics Role Labeling 53
6.3.2 Text Role Labeling 54
6.4 Feature Extraction 55
6.5 Bar Chart Data Recovery Accuracy 57
6.6 Summary generation 58

Chapter 7
Evaluation of Knowledge Base Construction **60**
7.1 Hyponymy Extraction 61
7.1.1 Correctness 61

7.1.2 Relevance 62
7.2 Taxonomy Evaluation 63
7.2.1 Correctness 64
7.2.2 Coverage 65
7.3 Comparisons with Open Information Extractors 66

Chapter 8
Conclusion and Future Work **69**
8.1 Summary and Conclusions 69
8.2 Future Work 70

Bibliography **72**

List of Figures

3.1 System Architecture 25
3.2 Role labels for Graphic and Text Components. 27

4.1 Semantic Representation of Bar Chart and its Extracted Features . 36

5.1 Knowledge base construction system 41
5.2 Graph insertions and merges for taxonomy construction. 48

7.1 KB entities that are frequently queried 66

List of Tables

5.1 Lexicosyntatic Hearst patterns 42

6.1 Chart-Data Extraction Precision, Recall and Type-II Error 51
6.2 Graphics Components Role Classification Accuracy using Multi-class C4.8 and RF Classifiers 54
6.3 Graphics Components Role Classification Accuracy using Binary C4.8 and RF Classifiers 54
6.4 Text Role Classification Accuracy using Multi-class C4.8 and RF Classifiers 55
6.5 Text Role Classification Accuracy using Binary C4.8 and RF Classifiers 55
6.6 Feature Extraction Evaluation Results 57
6.7 Data Extraction accuracy for Rule-Based vs. Machine Learning Role Labeling -1 58
6.8 Data Extraction accuracy for Rule-Based vs. Machine Learning Role Labeling -2 58
6.9 Generated Summary Evaluation 59

7.1 Comparison of taxonomy graph properties. 61
7.2 Relevance of Extracted Hyponymy Relationships 63
7.3 Annotator scores for extracted triple quality 63
7.4 Knowledge base coverage of top 3 million queries 63
7.5 Annotator scores for entity categories in knowledge base 64
7.6 Correctness and Relevance of Triples using NG-PL vs. OpenIE Systems on 100 Sentences 67
7.7 Hyponymy Extraction using NG-PL vs. OpenIE Systems on 100 Sentences 68

Acknowledgments

I wish to convey my gratitude to all who in any way aided me in this endeavor. Particular thanks are to the following:

To my parents, sisters, and brother for their endless encouragement and support.

To my advisor, Dr. Lee Giles whose advice, knowledge and guidance from the very early stages of research made this thesis possible.

To Dr. Jesse Barlow, for serving on my thesis committee and for all the support and mentorship since my first year in the program.

To Dr. Jeong Park, a colleague and sister, for her constant encouragement and support.

To members of the Intelligent Information Systems lab, especially Dr. Madian Khabsa, Dr. Kyle Williams and Kunho Kim, for the insightful discussions, helpful suggestions, and gracious camaraderie.

Finally, to all my friends and colleagues who made themselves available by providing support in many ways.

Dedication

I dedicate this thesis to the memory of my late grandmother (Jan. 2013), uncle Abdullah (Nov. 2016), and aunt Mzounne (Nov. 2016).

Chapter 1 Introduction

Typical search use for scholarly repositories such as Google Scholar[1], CiteseerX[2], SemanticScholar[3] and WebofScience[4], is via keyword-based queries that return ranked lists of links to scientific documents. In recent years, web search engines have introduced a new search paradigm, called entity-based search, which is mostly inspired by the semantic web community. Entity-based search abilities are typically made possible by use of knowledge graphs. Knowledge graphs model knowledge base information as entities and their relations. The nodes represent the entities and the set of labeled edges define the relationships between the entities. For web search engines, a knowledge graph is built from the contents of web pages by identifying explicit entities. i.e. semantic labels provided by the website, and implicit entities that are identified by contextual analysis of the text using natural language processing techniques. Existing ontologies and knowledge base data is used to further enrich the extracted entities and the relations among them.

For the scholarly search engine context, a similar approach can be applied since the scientific documents contain meta data such as title, keywords, and bibliographic information as well as content, which is the text of the document itself. Thus,

[1]scholar.google.com
[2]citeseerx.ist.psu.edu
[3]semanticscholar.org
[4]wokinfo.com

explicit entities can be easily derived from the metadata of the documents. The field of bibliometrics has extensively studied links and relations between bibliometric information including authors and publication data and has shown a vast amount of knowledge that can be extracted from the existing semantics provided with a document. However, if we wish to find an extraction analogous to the implicit entity identification, very few studies exist outside the biomedical domain that extracts biomedical entities. Thus, we formulate the implicit entity extraction problem for scholarly documents as a knowledge base construction problem. The purpose of this formulation is to explore the possibility for a *scientific* entity-based search realm.

As web pages contain multiple forms of data, scholarly articles also contain a variety of data formats such as images, tables, algorithms, charts in addition to the text. Thus, enhancing scientific search by processing queries in scholarly search engines as entity based queries rather than keywords based queries has merited potential. That is partly due to studies that have been able to extract and automatically analysis many of these forms of data in [1], [2], [3], [4], [5] enable semantic analysis of this auxiliary content of documents. Studies such as [6] provide an example of this semantic analysis by representing charts as related entities in a semantic graph.

To approach this problem, we define two tasks. The first involves ensuring the ability to extract facts when they exist not as raw text, but rather as auxiliary data, e.g. figures and tables. A system that is expected to automatically understand contents of scholarly documents, by definition, must be capable of understanding these forms of data as well. Scientific charts, such as bar charts, when used in scholarly documents almost always contain research results and findings, all of which are key facts about the study described in the document. Thus, by ensuring their content is included in the extraction process, the document-understanding claim is more accurate.

Scientific charts have wide presence not only as images in the web, but also as embedded figures in PDF documents. Main search engines nowadays include figures in search results. However, indexed content for charts, and documents containing them, rely mainly on the metadata textual tags. By not including the actual information these charts represent in the query process, search engines may overlook many valuable query results. Thus, enriching the indexing content for both documents and images based on the chart's content, provides an additional dimension to search improvement. Many studies aim to extract various types of information from scientific charts including bar charts. Automatically extracting the data values from bar charts can effectively enhance their semantic labeling. This additional information for charts can allow various analyses for search tools. Additionally, it can further assist the application of information retrieval and knowledge discovery techniques. We describe a system that targets bar charts specifically. The method automatically extracts text and graphical components from the charts and combines the results to infer the original data values of the chart.

Scientific charts contain valuable information for search engines. The metadata used to tag the chart image does not always provide enough information about the graphical contents of the chart. This has led many studies to address the problem of automating chart reading and understanding. Typically, the chart will represent data values that could be expressed as tables in a database. For a user, the values become known by visualizing the chart. However, for a computer to read the chart, image processing techniques must be applied to the chart along with some reasoning algorithm. Additionally, the designer of the chart may intend to illustrate a certain fact that the data values support, and this illustration is achieved by means of graphical effects in the chart. The purpose of this study is to propose a framework that is able to infer this fact from both the data values of the chart and the graphical effects employed in it's design by producing a corresponding summary in plain text. We focus on a specific class of charts, bar charts.

Summary generation from numerical data is a natural language problem. Linguistic studies propose constructs called protoforms that facilitate various forms of sentence generation. Protoforms generate linguistic summaries from data by exploiting the structure of the input data. Thus, in order to apply this method, the chart data is required to have some semantic structure. We propose to structure the chart data as a semantic graph. The appeal of a semantic-graph structure for the chart is due to it's wide applicability and ability to provide an abstract of the chart data. This abstraction of the chart data can be easily employed in various knowledge extraction and information retrieval applications. We provide an empirical evaluation of the proposed framework on a set of 300 bar charts extracted from scholarly PDF documents. The evaluation shows the results of key steps in the pipeline and the effects they propagate to subsequent steps. An overall evaluation of the summaries is conducted to verify the effectiveness of the system in automating the reading of bar charts.

In order for a knowledge base to be useful for scientific search purposes, it must have high accuracy of extraction and sufficient coverage of scientific concepts and entities contained in the documents. Thus, it is important to select an appropriate approach for KB construction that meets these needs. The main four approaches for KB construction are curated approaches, collaborative approaches, automated semi-structured approaches and automated unstructured approaches [7] . Curated and collaborative approaches are used for smaller domain specific KB since they require manual construction of the KB by expert or groups of experts, respectively. Automated approaches differ in the target text, as they can be from semi-structured texts or unstructured text. These typically use heuristics and rule-based approaches or machine learning and natural language processing techniques. Since The content of scholarly documents is unstructured, and the scale of the repositories dictates an automated approach, thus, for our model an automated unstructured approach is used. For automatic unstructured text approaches, harvesting entities and their relations methods can be classified into two categories: pattern-based methods or learning/reasoning methods [8]. Pattern based methods rely almost entirely on hand-crafted patterns to harvest new entity relationships whereas learning/reasoning method, use statistical analysis and natural language processing techniques to harvest new relations from existing ones. In our approach we use a pattern-based method to generate a seed of relations that are used for an iterative learning approach that harvests more entity relationships.

We address the problem of constructing a knowledge graph from a set of scholarly documents that provides more coverage of concepts found in scientific research content. Due to the large scale and diverse nature of scientific research content, the approach must be fully automated to avoid manually tagged seed data. Due to the specificity of topics in research content, find an existing knowledge base with sufficient entity coverage of research topics can be difficult. We propose using an iterative learning approach that bootstraps from a pattern-based method.

1.1 Motivation

In this section we discuss a brief background about each of the research tasks covered by this thesis. We describe main challenges in the problem of bar chart understanding for data at a large scale. This section also points out the limitations of significant approaches in the literature in order to set our problem apart from these works. We also describe the benefits of representing scientific content of research articles in a knowledge graph. This is followed by a discussion of the most prominent approaches in the literature of knowledge base construction.

1.1.1 Automatic Understanding of Bar Charts

The task of automatically reading scientific charts involves the process of transforming as much content of the chart from image representation to a structured text representation. This task has been the focus of many studies for several reasons:

- Scientific charts are widely available on the web as well as embedded images in documents. In the context of PDF scholarly documents, bar charts comprise almost 11% of the images found in papers published in top computer science conferences.

- In search engine algorithms queries retrieve results by matching texts against indexed content. Images are indexed using their meta-data and ranking algorithms measure their relevance to submitted queries by the relevance of the meta-data labels to the user query. The more intelligent the meta-data labels are the better the query results can be ranked. Automatic understanding chart content and transforming it to structured text format enables the enhancement of the semantic labelling of indexed charts. This directly improves search engine coverage and query results.

- Knowledge discovery and AI techniques have great advances and high quality results on text format of data. Transforming the charts to structured texts enables the application of these techniques to scientific charts.

- Applications designed for automatic reading of documents for the visually impaired must also be able to translate images, including scientific charts.

The problem of understanding chart images and representing them in a structured text format is typically approached in two ways. The first builds on the textual components of the charts, e.g., captions, chart title, axes names and scale and legend labels and then applying knowledge discovery and intelligent analysis to these extracted texts. This approach is more common in large scale applications where a large number of charts are processed. The other approach builds on extracting not only the textual components of the chart image, but deriving the original numerical data values that are represented by the combination of graphical and textual components in the chart. These however are more common in limited-size data sets. Although these approaches are useful and applicable each in their context, they are considered insufficient for application on large scale scholarly data repositories for several reasons:

- To the best of our knowledge, none of the approaches have been made publicly available as open source code at the time of this study.
- For large scale repositories where the data is used to index the charts and the embedding document, the first approach that relies on only the textual content of the chart, eliminates a large amount of meaningful information that can be used to describe the chart and document.
- Studies in both approaches have assumptions on the bar chart format that limit the percentage of charts available in digital format to which they can be applied. For instance, some proposed extractors focus only on binary images, others address only charts whose bars are all the same color (single-series data charts). Some address only charts that are extracted from PDF documents in raster format, or only vector graphics.

1.1.2 Scientific Knowledge Base Construction

Knowledge base construction is a problem motivated mainly by the semantic web. The semantic web promotes a proposal that machines can better understand web pages if their content is structured using semantic labels. Knowledge base construction is one of several approaches to semantic structuring. It is most present in studies that address the structuring problem at a large scale, e.g. the web.

Knowledge graph construction has been the focus of many studies for several reasons:

- Knowledge graphs enable entity-based queries to search engines. Major search engines such as Google and Microsoft Bing now use knowledge graphs as an integral component of the search engine. Microsoft Bing's entity based search result box, called a *Snapshot*, shows a combination of data facts in different formats of images, text snippets, and other data formats. This is enabled by using their knowledge base called Satori which contains over 22 billion entities and their attributes [9]. Google harvested a large knowledge graph from web pages. Google's Knowledge Graph [10] covers over 570 million concepts and over 18 billion facts about these entities. The KG is used to for entity based search and the results of the lookup in the graph are then integrated in the search results presented to the user.This also enables search engines to answer natural language-like questions in addition to simple look-up queries [7].

- Knowledge base representations of information are fundamental components for state-of-the art approaches that are currently used in key text understanding applications such as question answering. IBM have their own question answering system called Watson. At the core of Watson's question answering system is a knowledge base that it derives it's facts from [11].

- Knowledge graphs are important components in digital assistant systems such as Siri and Cortana.

- Automated summary generation systems such as the summaries provided in Bing Snapshots and Google's snippets rely on knowledge graphs to generate the summaries.

- Many decision support systems also use knowledge base data, as seen in the biomedical and life sciences domains.

In this thesis, we address the problem of constructing a knowledge graph from a set of scholarly documents that provides more coverage of concepts found in scientific research content. Although there exist many information extraction tools that extract triples from raw texts, these can not be directly applied to our knowledge base extraction system. Additionally, there are many approaches that generated well known knowledge base data sets, they are not fitted for our problem for several reasons:

- Many knowledge base studies make available their extracted knowledge base, however, to the best of our knowledge none of the approaches that extract hyponymy relations from unstructured texts have been made publicly available as open source code.
- Very few studies exist that evaluate the methods and algorithms used for knowledge base construction on a large scale scholarly repository, or from research papers.
- Most approaches that are applied at scale bootstrap on a knowledge drawn from an existing knowledge base. For scholarly repositories, most of these knowledge bases do not have sufficient coverage of research-specific terms.
- Domain specific knowledge base extraction studies that produced high precision mostly involve manually curated steps to guarantee extractions with better quality.
- Existing systems that address the problem of of knowledge base construction from scholarly articles as PDF documents do not provide a fully automated pipeline .

1.2 Objectives

The main objective of this thesis is to provide search engines query results with a source of intelligent indexing data collected from scholarly content, mainly chart images and article texts. The focus of the chart data extraction is to represent bar charts in digital format with intelligent semantic labels and meta-data. The aim is to achieve this goal by automating the entire process as well as ensuring it's applicability to large scale data sets. Since the data extraction is in the context of search engines and scholarly data repositories, the framework must be able to process standalone chart images in the web as well as chart images found in PDF documents. Thus our approach aims to extract the following data:

1. Chart text values
2. Chart original data values
3. Graphical properties used to emphasize or represent certain information

To meet these objectives it is necessary to evaluate existing methods and employ a combination of image processing and semantic representation methods. Additionally, it is key to illustrate the applicability of the intelligent semantics, and demonstrate by a well-known problem such as summary generation. This purpose of this step is to demonstrate how the use of intelligent semantic labels in NLP methods can enhance the meaningfulness of the resulting summaries.

To extract knowledge from scholarly document's texts, we focus on proposing a system to construct a knowledge base from the documents. Teh resulting knowledge base provides more coverage of concepts found in scientific research content. The knowledge base study in this thesis aims to acheive the following goals:

- Due to the large scale and diverse nature of scientific research content, the approach must be fully automated to avoid manually tagged seed-data.

- Due to the specificity of topics in research content, find an existing knowledge base with sufficient entity coverage of research topics can be difficult, thus the system must be capable of capturing scientific facts in the documents.

Since no ground truth for such a system exists, we also aim to explore knowledge base evaluation techniques that are capable of capturing the properties of the system.

1.3 Contribution

In order to meet our objectives we developed a framework to automate intelligent reading of scientific chart data, specifically bar charts, and provide the user with a textual summary of the chart. Additionally, we propose a system that reads scholarly articles in PDF format and constructs a knowledge base out of 10,000 documents. Our approaches have the following merits:

- We explore the accuracy of various image processing techniques in recovering the original data values represented by the bar chart.
- We investigate the correlation between certain graphical effects in the charts to specific designer-intended messages.
- We propose a semantic graph data structure to represent the chart information.
- We explore the feasibility of using the semantic graph as input to to fuzzy logic constructs called protoforms. in The summary generation module is responsible for reading the graph and applying the linguistic techniques to generate the summary.
- We deploy an NLP based system to extract entities and their relations from scientific texts in scholarly articles.
- We propose an iterative algorithm to expand the space of extracted entities.

- We propose a taxonomy construction algorithm to build a knowledge graph that covers over 15,000 computer science-related entities.

1.4 Thesis Organization

The remainder of this thesis is organized as follows. Chapter 2 presents definitions of image processing, semantic data, and fuzzy logic terms. It describes the significance of chart data extraction and main challenges in this domain. It also presents limitations of state-of-the-art chart processing studies. It also introduces concepts and methods used for summary generation. Chapter 3 explains our proposed method for extracting the data values from bar charts. It describes the supported formats and the assumptions on the input of this step, which is the input for the entire pipeline. It also provides the various output formats provided by the system. Chapter 4 describes the proposed semantic graph structure that is used to represent the set of charts. This chapter also explains the algorithm used to generate the summaries from the chart's graph representation. It presents the construction process of the various protoforms that are used in the summary generation algorithm. Chapter 5 Presents our knowledge base construction framework. It provides details on the iterative scientific entity learning algorithm as well as the taxonomy building algorithms. Chapters 6 and 7 present experimental results. These chapters describe thorough evaluations of each step of the proposed systems. Chapter 8 summarizes our research and provides conclusions and future research directions.

Chapter 2 Preliminaries

2.1 Chart Data Extraction

Various studies provide approaches to address the problem of understanding scientific graphics, algorithms, and tables in scholarly documents [12], [13], [1], [14], and [15]. Earlier work that focused on classification and data extraction of scientific charts in particular are [16] and [17]. More recently, [18] proposes a search engine called *DiagramFlyer* that indexes a large amount of scientific charts extracted from PDF documents. The method extracts the text from the charts and classifies their role, i.e. x-axis, y-axis, and uses it along with the figure metadata. Their method, described also in [19], does not show any module to extract the original data contained in the graphical components of the chart. Another approach for searching and retrieving charts is proposed by [20], which identifies certain structures that are extracted from charts and user-entered queries. A proposed linear model is then used to rank the charts whose information structures are best matched to those in the queries. One of the extracted structures is the notion of an *intended message*, which they describe in [21]. The method is based on the idea that a chart contains communicative signals which are extracted and used in a Bayesian network-based method to infer the intention of the bar chart. A further extension of this work is used to aid the visually impaired in reading graphical charts by means of an interactive chart summarizing tool called *SIGHT*, described in [22].

Any chart summarizing approach builds on a data extraction method that uses image processing techniques to extract the data values of the chart. The extraction method in [3] uses connected component analysis of the chart image. This is based on the method proposed by [23]. Other methods as in [13], target plot charts and apply machine learning techniques to extract the data. Some methods are based on an analysis of gray level histograms, such as the one proposed by [24].

2.1.1 Chart Figures in Scholarly Documents

Digital libraries, on which academic search engines are built, store scholarly articles in Portable Document Format (PDF). This format allows for embedding figures, tables and other artifacts in a variety of formats. Chart figures, the interest of this work, are typically embedded in PDF files using one of two different formats: vector graphics and raster images.

Raster image format is the a bit-by-bit representation of pixels in an image, also called bitmap. The image can be seen as a matrix of pixel colors. Raster graphic images are stored in compressed file formats such as GIF, JPEG, and PNG. Raster figures embedded in a PDF documents, are typically extracted by converting the PDF document to a bitmap image and then perform document image segmentation. The images are returned in any of the above compressed file formats, with varying qualities of resolution depending on the rendering and original document properties.

vector graphics, as the name indicates, uses vectors to store the data points of a set of polygons representing the image. The standard format for vector graphics is called SVG short for Scalable Vector Graphics, which is much smaller is size than raster image files. The SVG graphic can be rendered at time of use, either display or printing, by converting it to raster format at any scale without any loss in resolution. It is most commonly used for representing text, but is also for digitally created graphics, such as charts, maps that are not photographs. Common vector graphic extractions from PDF documents are done by identifying the path objects in the graphic component found in the PDF stream. The path objects are then converted to to SVG path objects and the final image is extracted as an SVG file [25].

2.1.2 Connected Component Analysis

Connected component labeling is an image processing technique used to identify objects in an image [26]. This is achieved by a scan of the pixels of the image and assigning a label to each contiguous set of pixels. If a set of pixels have similar intensity values, that indicates that they belong to the same graphical object.

Connected component labeling is most commonly applied to binary and grey-scale images, but it also can be applied to colored images. When scanning the image and assigning labels based on pixel connectivity, neighborhoods of 4 pixels or 8 pixels are considered. Generally, connected component labeling follows these main steps:

- Beginning at the top leftmost pixel, scan the image row-wise and assign a label to the pixel based on whether it is similar to its neighboring pixels or not.
- If a pixel is similar to it's neighbors it is assigned the same label. If it has a different color, then a new label value is created and assigned to the pixel.
- If the pixel is similar to any of the neighbors, yet the neighbors have different label values, assign any of the labels to this pixel and record the equivalence of labels in the neighborhood of the current pixel.
- After the first pass over the image, consolidate equivalent labels into one label with a single label value.

- Run another pass over the image and replace each label with the updated label value from the previous step.

As evident from the steps above, the case for binary images is much simpler than colored images. With colored images, the pixel color must be compared with all other pixels, as there are more classes than black or white.

In order to compare the color of a pixel for connected component labeling, many color scales are used in computer vision applications. RGB color space is the most commonly used color scheme for raster graphics. Comparing similarity between pixels when represented in RGB values requires the existence of a measure of difference between RGB values that gives a high numerical value for visually different colors and low values for visually similar colors. Since RGB values for colors do not reflect the colors as perceived by humans, no such measure can be used. As an alternative, other color spaces are used such as the *Lab* color scheme. This color scheme mathematically describes colors using three values per color: lightness, and green-red and blue-yellow color opponents. The L is for the first, and a and b are for the second two values. Uniform changes in the three values numerically correspond to uniform changes in the perceived color, thus allowing for a metric of color similarity that captures the differences in colors as a human eye would. A formula called DeltaE94, or dE94, computes the difference between two Lab colors based on the three factors of lightness, chroma and hue value.

$$dE94 = \sqrt{\left(\frac{\Delta L}{k_L S_L}\right)^2 + \left(\frac{\Delta C_{ab}}{k_C S_C}\right)^2 + \left(\frac{\Delta H_{ab}}{k_H S_H}\right)^2}$$

where,

$$\Delta L = L_1 - L_2,$$

$$\Delta C_{ab} = C_1 - C_2,$$

$$C_1 = \sqrt{a_1^2 + b_1^2},$$

$$C_2 = \sqrt{a_2^2 + b_2^2},$$

$$\Delta H_{ab} = \sqrt{\Delta E_{ab}^2 - \Delta L^2 - \Delta C_{ab}^2},$$

$$S_L = 1,$$

$$S_C = 1 + K_1 C_1,$$

$$S_H = 1 + K_2 C_1,$$

such that K_1, K_2 and k_L depend on the application, k_H and k_C unity values. The computed difference using deltaE94 between two colors is used to compare pixels to their neighbors in order to determine their labels. The more reflective the measure is to the colors difference, the better the result of the connected component analysis.

2.2 Summary Generation

The problem of generating linguistic summaries from charts has been addressed by many projects, such as the iGRAPH-Lite project [27] that presents the data facts contained in the chart as a textual summary. Another summarization system is the one proposed by [28], where they provide the intended message, described above, as a summary. Summarizing numerical data and time series is also the focus of various studies [29], [30]. An example is the method for generating summaries from data collected by medical-purpose sensors by [29]. The method generates the summary of eldercare data using protoforms, a concept proposed by [30].

Since the summary application at hand is for data that is extracted from a bar chart as a data table, we focus on linguistic summary techniques that are designed for numerical data. An approach by Yager [31] and Zadeh et al. [30] generates summaries in the form of several natural-language sentences that capture some properties descriptive of the numerical data as a whole. The linguistic summary model is defined on a database D that contains a set Y=$\{y_1, ..., y_n\}$ of objects. Objects from Y are described using a set of attributes A=$\{A_1, ..., A_m\}$. The linguistic summary contains the following components:

Summarizer, P This is a coupling of an attribute A_j with a linguistic fuzzy predicate, e.g. low, high, etc.

Quantity in agreement, Q A linguistic quantifier such as some, half, most, etc.

Qualifier, Q An optional additional attribute coupled with a predicate over an attribute A_k that describes a subset of Y.

The components above are combined in a linguistic construct called protoforms which are in the form:

$$Q\ y\text{'s are } P$$
$$QR\ y\text{'s are } P$$

An example of a resulting sentence is "Most cars with color red have highest sales". Here the summarizer P is a combination of the attribute *sales* and the predicate *highest*, Q is the quantifier *most* and R is the qualifier *cars with color red*.

2.3 Knowledge Base Construction

Knowledge graph construction from large scale corpora has been the focus of many studies. Examples include Google Knowledge Graph [10] , NELL [32] , YAGO [33] and DBpedia [34]. Some KBs are constructed using manual curation, such as the Cyc project [35] that has a limited concept space coverage of only 120,000 concepts. To improve coverage from a curated method, collaborative crowd-sourcing was used for the construction KBs such as Freebase [36] and Wikidata [37]. These have shown better coverage results however, they do not contain scientific concepts that are at a low level of granularity in specific scientific fields. More automated approaches for extraction include the NELL [32] , KnowItAll [38], ReVerb [39]. These do contain many concepts that may exist in scientific texts yet they are general concepts. A more comprehensive KB is that contains finer-level concepts is Probase [40]. Probase is constructed from a corpus of 1.68 billion web pages and harvested over 2.7M concepts. Although concepts are less course than other KBs, it's corpus is web pages rather than research articles and scholarly content in general.

A hyponymy relationship between two entities occurs when one of the entities is an instance of the other, also referred to as the *is-a* relationship. The super-concept is called *hypernym* whereas the instance, or sub-concept is called the *hyponym*. Many studies aim to extract hypernym-hyponym pairs using various approaches. One of the initial works is [41] that introduced the Hearst patterns used by most other approaches for bootstrapping. Latent semantic analysis is used to improve the results obtained by using the Hearst patterns by using latent semantic indexing for the hypernym-hyponym terms pair [42]. It then compares the similarity score between the two to infer if one of the terms is actually an instance of the other. In [43] they use a training set of known hypernym-hyponym pairs to train a model using the dependency path features. The model then learns new dependency paths and uses them to extract new *is-a* relation pairs.

Iterative approaches for knowledge base extraction such as TextRunner [44], NELL [32], and Probase [40] use a bootstrapping approach. The first two iteratively learn new syntactic patterns to improve the extraction for the next iterations. Our approach is similar to the one in [40] in that our iterations learn more entities and relations by using knowledge extracted from previous iterations.

Automatic taxonomy construction from raw text in some cases makes use of an existing knowledge source. An example of these works is [45], where they construct a taxonomy using external knowledge such as Wordnet [46] and Hearst-style patterns.

In other studies the approach itself does not rely on external knowledge sources to construct a taxonomy for web data , such as YAGO [33] and WikiTaxonomy [47]. Although, our method is similar to these in that we do not use external knowledge sources, however, they construct the taxonomies from Wikipedia articles. Other studies provide approaches for merging the nodes for combining taxonomies, such as [48], where word embeddings are used to determine the similarity between two taxonomies in order to merge them.

Chapter 3
Bar Chart Data Extraction

In this chapter we describe the methods used to extract data from bar charts that are found in scholarly articles. First, a rule-based method is proposed and described. Next, a machine learning-based approach is proposed and also described.

3.1 Rule-Based Data Extraction

Our rule-based method follows the pipeline shown in Figure 5.1. The system comprises of three main modules: graphical-component extraction module, text extraction module, and the data inference module. The system takes a 2-dimensional bar chart in image format as input and regenerates the chart as output. The following assumptions are made regarding the charts:

- Charts are 2-dimensional.
- The fill color for the bars is a solid color, rather than a pattern.
- The y-axis follows a linear scale, rather than logarithmic.
- Y-axis alignment is to the left, x-axis alignment is to the bottom of the chart.

Following these assumptions the system modules extract bar data automatically. The system initially applies some pre-processing steps to the chart image, such as color normalization and noise filtering. Algorithms for text recognition in graphics typically apply noise removal filters to the image first [49]. For scientific charts, however, it is uncommon to have highly noisy images [50] as compared to non-chart images. Additionally, noise removal filters, e.g. bilateral filter, while computationally costly, provided no significant improvements when applied to charts that were extracted from PDF documents. Thus, we do not apply any noise filtering step. The only pre-processing step in our method is changing the color-space prior to passing the chart to the extraction modules.The text extraction module requires a grayscale version of the chart. For the graphical component extraction the image is converted to the Lab color space. This is due to the fact that our method is expected to apply to colored charts just as effectively as black and white ones. The images in PDF documents are recovered in raster format, where each pixel is specified by its RGB value. Processing a chart in the RGB color scale proposes the following challenges:

- A single bar can have very light color that is close in RGB value to that of the background. This makes it less distinguishable as a distinct graphical component in the chart.
- In multiple-series bar charts where bars can share an adjacent edge, adjacent bars can be mistaken to be the same graphical component if their RGB colors do not highly contrast.

These challenges are far less present when the recognition of components is applied to in image in the LAB color space. The LAB color space is more perceptually uniform than the RGB space. This implies that the amount of change in the color values corresponds to a similar amount of change in human perception. This makes it effective in extracting information that is distinguishable based on visual effects, such as chart components. In order to extract meaningful information from the chart, two basic tasks must be performed. First, extraction of the graphical components, i.e. the bars and axes. The other task is the retrieval of texts that label these graphical components. Additional processing of the extracted information is required to obtain the data values of the chart. The methods proposed to achieve all these tasks are described in the remainder of this section.

3.1.1 Graphical Component Extraction

Once the bar is converted to the Lab space, the graphical component extraction module is responsible for identifying the main chart component, the bars. For accurate data extraction, the bar extraction method must first: correctly identify each bar distinctly, and secondly have high recall, i.e. identify as many bars as possible (preferably all of them). In order to recover the bars, our algorithm follows a similar approach to the one presented in [23]. We perform connected component labeling to the image, however our method uses the Lab color space to distinguish components. For the connected component method, neighboring pixels who differ in color are labeled as different components. To compare colors in the Lab space we use the delta-E 95 distance equation. Upon experimentation the threshold of 7 was found suitable. Once the connected components are recovered, we identify the bar components by using heuristics derived from the graphical properties of the bars. The following properties are used to distinguish a bar component:

- A bar fills its bounding box with ratio greater than 90%.
- The color of any pixels inside the bar is different than the color of all pixels within 2 to 3 pixels distance outside the bar edges.

Each bar is defined by its location and height in pixels. In order to identify the x-axis we use the conjecture that the x-axis is the horizontal line that all bars have an edge on. We use a histogram of the location of the base of the bars to recover a common horizontal coordinate among the horizontal edges of all the bars. This is defined as the x-axis.

3.1.2 Text Component Extraction

The texts associated with the graphical components of the chart are key elements to recover the original values represented by the chart components. Text components include: labels for x and y-axes, and each of the bars, in addition to the numerical values marking the scale of the y-axis. In this section we describe our proposed method to automatically extract these pieces of valuable information. The method involves two basic steps as following the method in [49]. The first step is to automatically locate text regions in the image. The next step is to apply OCR recognition to these text components.

3.1.2.1 Identification of Text Regions in a Chart

To identify the text regions in a chart we follow similar steps to the method proposed in [49]. The image is already binarized in step A above. We remove components whose area size is greater than that of a typical character size. Next, in order to identify which letters represent a single word, the letters are subjected to an isotropic dilation with a small window size. This will close the small gaps between pixels that are close to one another just enough to be adjacent. Then we apply connected component labeling to the dilated image. This labeling will label entire words as one component. To maintain image quality for the OCR, once the locations of the text regions are specified, the text region blocks passed to the OCR step are from the image before dilation rather than the dilated ones. If a text region's width is smaller than its height, it is vertical and most likely to be the name of the y-axis.

3.1.2.2 Extracting Text and Numerical Values

The tesseract OCR is used to recognize the nominal or numerical values of each text region. The results are filtered out for any text regions that produce empty spaces or only punctuation marks. For instance, some stray pixels may be identified as texts and return the ". character. Finally, the recovered texts are represented as their nominal or numerical values along with their locations.

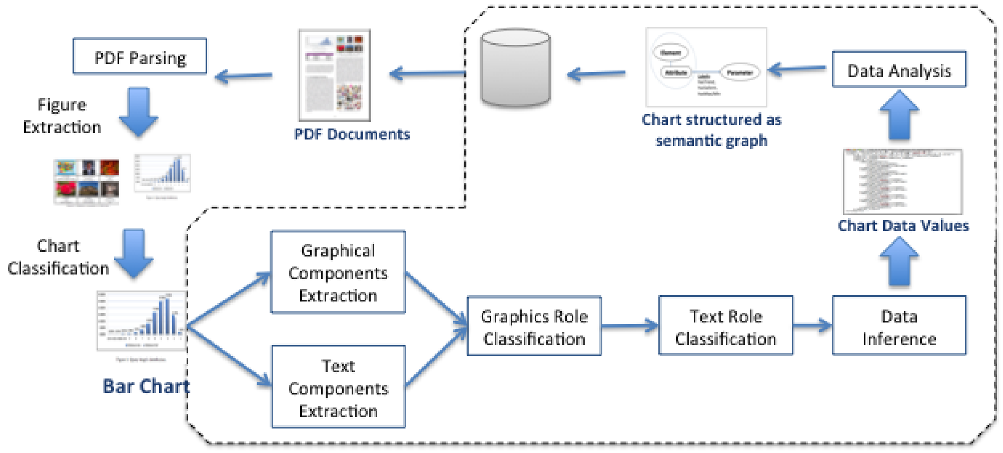


Figure 3.1: System Architecture

3.2 Machine Learning based Extraction

The rule based method faced several challenges. The first is regarding the y-axis detection. The scale of the y-axis is key for accurate extraction of the data values, thus the precision obtained by a rule-based method is not sufficient. In order to improve the precision we propose a machine learning approach that is described in this section. the data values represented by a bar chart, the chart text and graphics components must be identified and their locations retrieved. This section covers the techniques and methods used for component extraction. The extraction process follows the pipeline illustrated in figure 3.1. The first step is to extract two types of components, the graphic components and text components. Graphic components include: x-axis and y-axis, chart legend, and bars. By text component we refer to all text labels found in the chart area. The graphic components are extracted using the method described in [3].

3.2.1 Graphic Component Extraction

The method applied to extract graphic components follows a 3 step process:

Image Color Space Conversion The image color is converted from RGB space to the L*ab space. This step is performed to provide higher accuracies in distinguishing a wider spectrum of colors than can be achieved using RGB color space.

Grid Line Removal Hough transforms are used to identify horizontal lines that are not long enough to be the x-axis. This step is useful in eliminating background boxes.

Color Connected Component Labeling This is performed over the image in L*ab color. The distance function used to measure the difference between pixel colors is the deltaE95 function. This step returns, for each of the connected components, the bounding box coordinates, area, and color.

3.2.2 Text Component Extraction

The text region extraction steps are as follows:

Binarization The image is converted to binary format.

B/W Connected Component Labeling A pass of the black and white connected component labeling is run on the binary image. The returned components are then filtered to remove large components that are not candidates for being text characters. This step returns regions for single characters in the image.

Isotropic Dilation Each extracted character is dilated, using a small enough window, only to make each character attach to the character next to it if they are in the same word. The window size is chosen to be small enough to not reach a standard space size.

2nd Pass Connected Component Labeling Since the previous dilation is performed, each word is now a single component rather than each character. Thus, the output of this pass is the coordinates of the bounding box of each text word in the chart image.

Text Recognition-OCR The patch of the image where the text region is located is passed to an OCR tool that returns the text content of that part of the image in string format.

The output of the text region extraction step is the location of the text region and the text string value. This is then passed to the text role labeling step.

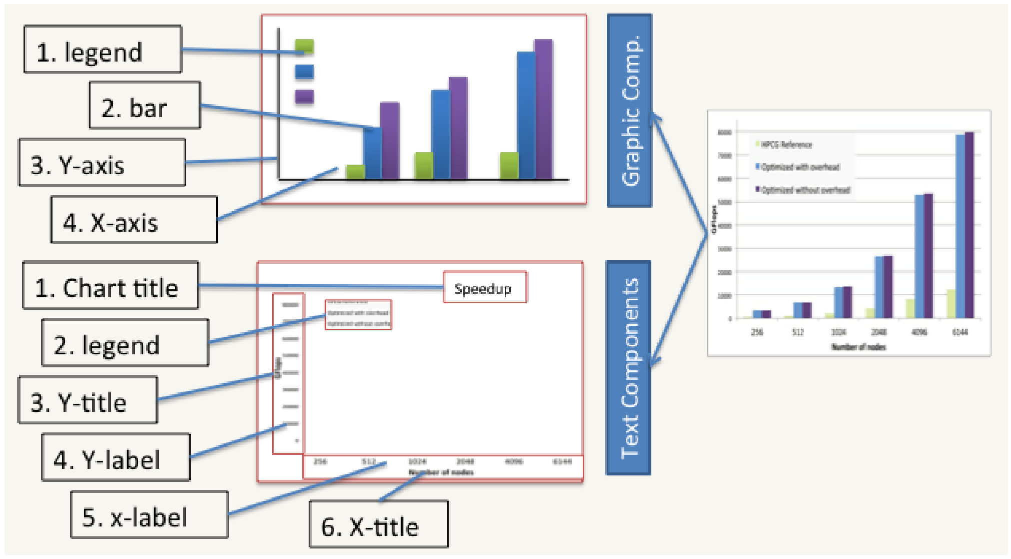


Figure 3.2: Role labels for Graphic and Text Components.

3.2.3 Component Role Classification

From the previous section, the chart has been reduced to a set of image components. The information we have about these components at this point is the location of their bounding box in the image, whether they are text or graphics, color for graphic components and text string for text components. The next step to obtain the data is to correlate the text descriptors and numerical values with the graphic component whose value they are describing. This process is referred to as *role labeling* or role classification. Figure 3.2 shows the different role labels defined by our system for both graphic and text components. (Sample chart is from [51]). In this section we describe the methods used for role classification.

3.2.3.1 Graphic Component Classification

For bar charts, we define four role labels, bar, legend, x-axis, and y-axis. The components are subject to a noise cleaning step to remove false positives. We define 10 features for the graphic components. The graphic features that are selected for these components are:

Shape We are interested in rectangular/square shapes for bars and legends. The extent of a shape is a value between [0,1] that measures how much a shape fills its bounding box. The higher this value, the more likely it is a box shape. This feature is for both bars and legends, as both are box shaped.

Color The color of the centroid of the shape is used for this feature. The value is a binary true if the shape is either black or white and false otherwise. This feature determines background boxes.

Distance to X-axis The value of this feature is a the distance between bottom edge of the bounding box and the x-axis normalized over the image size. For bars, the distance will be very small or zero. For legends it can vary.

Relative position to Y-axis This takes a binary value of true if the shape is to the right of the y-axis. Bars are always to the right of the y-axis. Legends may or may not be.

Relative shape width The relative width of the shape to mean width of all graphic components in the image. Legends are usually smaller or wider than bars. Also, small sized bars may have their total area similar to that of a legend box, but the difference is that it's width is similar to those of other bars.

Centrality This is the normalized distance of the component from each bisector of the image. This is for legends as they are typically closer to the borders of the image.

Height-width ratio The ratio of height to width of the shape's bounding box. This feature is for legends since they are more commonly square-shaped.

Type of closest component This feature takes a binary value of true if the closest component to the graphics is a text component. This feature is also for legends since they are typically closest to texts.

We use the features to classify the role of the graphic component into either bar or legend. The classification results are tested using the c4.8 and random forest methods to compare accuracy levels obtained by each classifier.

3.2.3.2 Text Region Classification

For the text components we extract the features from the graphic properties of the text regions. Additionally, we also have one feature that is related to the actual text value of the text region. The classification is to determine what graphic component the text is describing. The approach we use is a combination of two approaches. We propose three types of features for the text role classification: location/position-centric, text-centric and graphical features. The location-based features are an adaptation of the classification method proposed by [16] where they specify 5 features that are extracted from the texts to classify the regions. The text-centric features are adopted from the [19] method. Graphical features are similar to those used in the heuristics-based methods of [3] and [23]. The texts are classified into one of 7 roles: y-axis label, y-axis name, x-axis label, x-axis name, legend name, chart title, and other. The location/position based features mentioned are the following:

Distance to closest graphics The distance between the text region and the closest graphic component to it, is determined as following:

$$h(T_i, G_j) = \underset{t \in T_i}{min}\ \underset{g \in G_j}{min}\ d(t, g)$$

where, T_i and G_i are the text and graphic components respectively, and t and g are the sets of points on the perimeter of each component. This is a measure of the closest Euclidean distance, d, between the two closest points on the perimeter of each component.

Relative position of the closest graphics This is determined by the angle between a text block and it's nearest graphic component. The positions take values: top, bottom, right, left. top right, top left, bottom left, bottom right.

Position to Y-axis This is a binary value that is true if the text is to the left of the y-axis.

Position to X-axis Also, a binary value. The value is true if the text region is located below the x-axis.

Centricity Horizontal and vertical centricity are calculated as the normalized distances between the centroid of the text region and both the vertical and horizontal bisections of the image.

The text-centric features are the following:

Capitalization Percentage of characters in the word that are capitalized.

String Length Normalized number of words in the text region. Axes titles and chart title are more likely to contain more than one word.

isNumeric Whether the text has a nominal or numeric value. The y-axis labels are always numbers.

The following are the graphical based features:

Orientation Vertical or horizontal orientations of the text box as a binary value. This feature is for the y-axis title, which typically appears in charts in a vertical layout.

Closest Graphic This feature stores the class of the graphic component closest to the text box. This requires that the graphic component classification has already been completed.

These features are then used to train both a c4.8 and a random forest classifier to determine the role of the text label.

3.3 Chart Data Inference

In this step we extract the chart data by applying an inference process. The name of the bar is the text identifying the bar, which is typically located under the bar. The value of the bar is the y-axis value that corresponds to the highest point of the bar. To extract this data pair we follow steps similar to those in [23]. Three main values must be recovered to accurately obtain the bar values. These are: the y-axis scale values, the data-per-pixel ratio, and the bar name, i.e. x values. Algorithm 1 shows the steps of the inference method. As input, it requires the location of the x-axis and y-axis, which have been previously located, the set of bars B, and the extracted text strings T. The y-axis is determined to be the area left to the left-most bar. The x-axis location has been extracted in the previous

```
Input   : x-axis, y-axis, B, T
Output : Data values ($data.name, data.value$),
         $Y\_name$, $X\_name$
if $T_i.y$ below x-axis then
    $X \leftarrow T_i$
end
Assign elements of $X$ to $X\_name$ and $X\_labels$
based on horizontal alignment
if $T_i.x$ to left of y-axis then
    $Y \leftarrow T_i$
end
Assign elements of $Y$ to $Y\_name$ and $Y\_labels$
based on vertical alignment
for $T_i \in Y\_labels$ do
    $Scale_i = T_{i+1}.y$ - $T_i.y$ /$T_{i+1}.value$ - $T_i.value$
end
$\eta = median(Scale)$;
Set $data_i.value \leftarrow B_i.height * \eta$;
Set $data_i.name \leftarrow X\_labels_i$ where:;
$X\_labels_i.x$ is closest to $B_i.x$
return  : set of data pairs ($data.name, data.value$)
```

Algorithm 1: Data inference algorithm for extraction of chart numerical values and axes names.

graphical component extraction step. Each graphical component in B_i contains 3 values. The x and y coordinates and the height of the bar, denoted $B_i.x$, $B_i.y$, and $B_i.height$, respectively. The text strings in T also contain 3 values. The x and y coordinates of the text region and the value of the string, denoted $T_i.x$, $T_i.y$, and $T_i.value$, respectively. The first value to infer, the y-scale, relies on both the correct location of the texts describing the scale and the result of the OCR. To specify the texts that correspond to this values, we apply two heuristics: the y-axis labels are to left of the leftmost bar, and the are vertically aligned. Those texts that satisfy these assumptions are sorted and used for the next step. The next step is identifying the pixel-per-data ratio. For this, we calculate the difference between the y-labels in pixel and divide it by the difference in the numerical values of the y-label texts. The bar values are thus, their heights in pixels multiplied by the pixel-per-data ratio. The Nominal value for each bar is the value of the text located below the bar. To identify the values we extract the text regions who are located below the x-axis and are horizontally aligned.

Chapter 4
Semantic Graph Representation

The purpose of chart component extraction is to convert the representation of information contained in a chart image from graphical to a text representation. Depending on the application requiring the chart information, the needed information representation can vary. In this section we describe how we produce machine-usable information from the data extracted from bar chart images.

4.1 Data Analysis

The first step towards understanding a chart's content is to extract the original data values from the graphical component. The method described in [3] is used to extract the following data values from the bar chart: (1) x and y axes names, if they exist, (2) numerical value for each bar (the y value), (3) nominal value for each bar (name of the data value the bar represents). Once the data is recovered from the chart image, it is then passed to the feature extraction method. The graphical features are special graphical effects used by the designer of the chart to illustrate a certain fact, or as by [20] refer to is, the message of the chart. Based on empirical analysis of charts messages performed in [20], the most common messages charts typically aim to convey, are:

- Increasing/decreasing trend.

- Showing the rank of each value in the chart.
- Drawing attention to the rank of a specific value, or subset of values.
- Showing the maximum value among a set of values.

Other messages include the minimum value as well, where the percentage of charts expressing this message type was found to be less than 15% [20]. In this work we select messages that are more prominent among bar charts. Each of these messages has certain graphical features in the chart associated with it. Arranging the bar data in increasing or decreasing order for the bars can be considered an indicator that the chart is illustrating a trend. A bar that has a color different than all other bars in a single-series chart, indicates that the bar may be of specific importance to the meaning of the chart. Other annotations, such as labeling only certain bars with their value or other textual data suggests the designer would like to draw attention to their values. In [20], they refer to these as *Salient* bars. If none of the bars in the chart image have any salient elements, nor display a visible trend, it is most likely that the purpose of the chart is to simply display the values and how they rank to one another. In [20] the message of such a chart is referred to as communicating the bars' *Rank*. Based on the findings mentioned above, we select three type of features to extract from the chart data: Salience, Trend, and Rank. Bars are marked Salient if they have a distinct color or are annotated with texts. If the bar values are monotonically increasing, or decreasing, this indicates the presence of the Trend feature. Additionally, if the x-axis labels are a time-series or have ordinal values, this is a feature labeling the data series as a whole. As for the Rank feature, we provide the user with only the maximum and minimum values found in the chart for simplicity. Once the extraction of the data and the features is complete, the next step is to generate a semantic-graph to represent the data. This process is described in the next section.

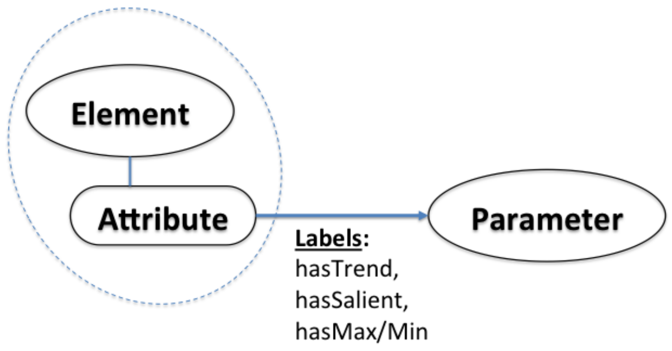


Figure 4.1: Semantic Representation of Bar Chart and its Extracted Features

4.2 Semantic Graph Construction

Constructing a semantic graph requires definition of relevant semantics. Semantic labels are derived from the roles of each extracted data value or feature. The y-axis name has semantic label *element*. If the y-axis name contains a noun and a descriptor, the descriptor is labeled *attribute*. The x-axis name has semantic label *parameter*. The edge between the attribute (or Element of no attribute was found) is labeled with the features extracted by analyzing the data. Three labels are used for edges. *hasTrend* can be: increasing, decreasing, or none. *hasSalient* determines whether any of the bars are salient, either by annotated text or a different color. *hasMaxMin* specifies the maximum value of the bars and the minimum . This is used to to describe the chart in case no other features are present.

4.3 Automatic Summary Generation

To generate the summary we follow a method similar to [29] which is an adaptation of the concepts initially proposed in [30]. The method is based on generating linguistic summaries from protoforms. A protoform is a linguistic construct that allows the use of structured data to generate English language sentences using fuzzy logic.

4.3.1 Fuzzy Logic Constructs

We define 4 basic protoforms for each type of feature the relationship between the *element* node and *parameter* node may share. The protoforms are: trend protoform for a time series, trend protoform for an ordinal data set, salience protoform, and max/min protoform. We use notation similar to that in [29], with some modifications to fit our context:

- y is the *parameter*.
- Q_i is the specific value that the *parameter* may take.
- P a summarizer, which is a tuple made up of the subject of description, *element* and a fuzzy predicate, e.g., low, high.

The predicates we use to pair with the element to make up the summarizer P, are the following: increasing, decreasing, highest, and lowest. The first protoform is for a trend in a time series:

$$Q_{min} \text{to } Q_{max} \text{has } P \tag{4.1}$$

where Q_{min} and Q_{max} are the beginning and end of the time series represented on the x-axis. The predicate values that are used in this protoform are increasing and decreasing.

The next protoform is a trend for an ordinal series for the x-axis:

$$(y, p) \text{ has } P \tag{4.2}$$

where y is the parameter that has an ordinal value and p is the trend of the ordinal value, which can be increasing or decreasing.

The next protoform is salience protoform:

$$y\, Q_i \text{ has rank } P \tag{4.3}$$

where the predicates of P are xth highest or lowest.

The last protoform is the max/min protoform:

$$y\, Q_i \text{ has } P \tag{4.4}$$

Here, the predicates are highest and lowest for the maximum and minimum values, respectively.

```
Input  : Semantic Graph $G$
Output : Summary Sentence $S$
if $hasTrend$ is not nil then
    $S$ = protoform 1 for time series
    $S$ = protoform 2 for ordinal series.
    $S$ = nil otherwise.
end
if $hasSalient = TRUE$ then
    $S = S+$ protoform 3
end
if $hasSalient = FALSE$ and $hasTrend = nil$ then
    $S$ = protoform 4
end
return : $S$
```

Algorithm 2: Algorithm to Generate Summary from a Chart's Semantic-Graph

4.3.2 Summary Generation Algorithm

To generate the summary, the method follows the steps illustrated in Algorithm 2. As input, the method takes a semantic-graph representation of the chart and returns a text summary in the form of sentences stored in string S. Based on reading the labels associated with edges between the element and parameter nodes, the algorithm determines which protoform to use for generating the summary. One or more sentences can be generated and eventually combined to produce the final summary.

Chapter 5 Scientific Knowledge Base Construction

In this section we provide a formal definition to our extraction problem and a description of the system pipeline.

5.1 Problem Definition

In this paper we design a system to extract related entities from a set of scholarly documents. The entities are nouns and noun phrases and the type of relationship we extract is the hyponymy relation among entities. A hypernym-hyponym relationship is one where one entities is an instance of the concept represented by the other. This is also known as the *is-a* relationship. We define hyponymy relations to occur between 1) a *concept* and *sub-concept* or 2) between a *concept* and an *instance*. We follow the same formal notation used in most hyponymy extraction studies [42], [40]. For a document set D, we extract a set of sentences S that contain candidate tuples (X_i, Y_j) such that,

$$\text{s} = \{(\text{X}, Y_1), (\text{X}, Y_2), ..., (\text{X}, Y_k)\}$$

where X is a hypernym for candidate hyponyms Y_j. Once the extraction is complete we have a set of extracted pairs,

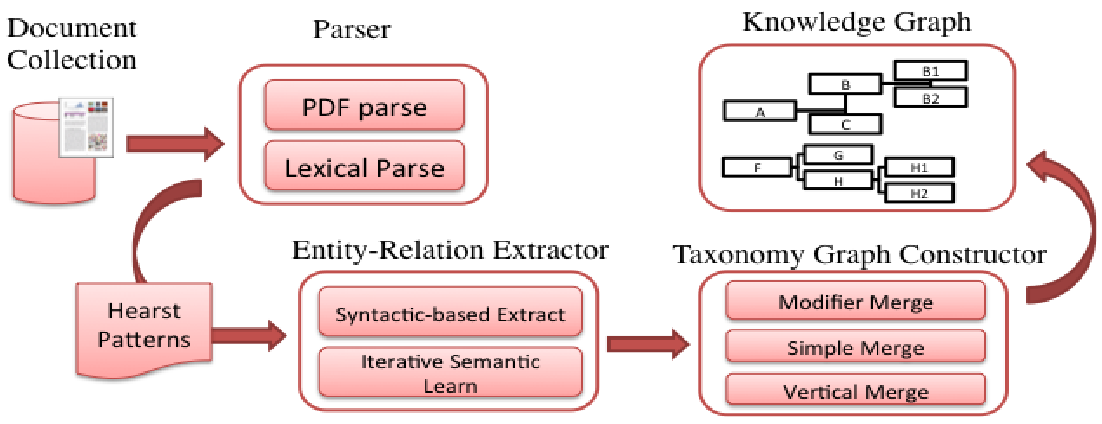


Figure 5.1: Knowledge base construction system

$$P = \{(X_1, Y_1), (X_2, Y_2), ..., (X_N, Y_N)\}$$

The knowledge base is constructed by building a taxonomy T by combining the local taxonomies identified in by the triples in the set P. We define the sets X=$\{x_1, \dots ,x_m\}$ and Y=$\{y_1, \dots ,y_n\}$, with $|X|$=m and $|Y|$=n, to be the sets of individual words comprising the noun phrase of a candidate hypernym and hyponym, respectively.

5.2 System Overview

The pipeline of our system is shown in Figure 5.1. Our system is given a set of documents in PDF format, and generates a knowledge base represented as a graph. The system is comprised of three main modules: a parser, an entity-relation extractor, and a taxonomy graph constructor. Below is a description of each module.

`Parser` The first module in the pipeline is the parser. This is responsible for two types of parsing. The first is a rendering parser, which extracts the raw text from the PDF file. For this task we use the GROBID [52] parser. The second parser is a syntactic parser that extracts dependency-paths to identify noun phrases. This module passes a set of parsed sentences to the next phase.

Table 5.1: Lexicosyntatic Hearst patterns

#	Pattern
1	*NP* such as *NP* (, *NP* and/or *NP*)
2	such *NP* as *NP* (,*NP* and/or *NP*)
3	*NP*, including *NP* (,*NP* and/or *NP*)
4	*NP* (, *NP*) and other *NP*
5	*NP* (, *NP*) or other *NP*
6	*NP*, especially *NP* (, *NP* and/or *NP*)

`Entity-relation Extractor` This module has two basic components. The first is a syntactic based extractor that takes the set of pre-defined Hearst patterns as a basis for matching. The syntactic-based extractor, which generates the set S of sentences containing candidate *is-a* triples. The next component is the iterative semantic learner, which iteratively learns new triples from the set S and populates the set P of the extracted pairs of the knowledge base. This step passes a set of hypernym-hyponym triples to the taxonomy construction phase.

`Taxonomy Graph Constructor` Taxonomy construction is accomplished through three main graph operations. The set of extracted triples are initially mapped to local taxonomy graphs. Next, based on the existence of similarities among nodes of distinct local taxonomies, we apply modifier, simple, and vertical node merging operations to generate the final knowledge graph G.

5.3 Hypernym Relation Extraction

In this section we describe the two modules responsible for generating the knowledge base tuples of entities related with a hypernym-hyponym relation. The input to this step is a set of PDF documents D and the output is the set P of pairs (X,Y).

5.3.1 Syntactic-based Extraction

Since our approach does not rely on an existing knowledge base, we bootstrap our algorithm by extracting pairs using a set of hand-crafted patterns. The use of lexico-syntactic patterns for bootstrapping hyponymy extraction is a common practice since they have relatively high precision while compromising recall, which is tolerable for a bootstrapping step. We use the Hearst patterns from [41] that define syntactic patterns that are commonly used to denote a hyponymy relationship. Table 5.1 shows the patterns used in our system. NP stands for *Noun Phrase*, that are first identified and then matched against the pattern. The Stanford lexical parser [53] parses the sentence to extract noun phrases. These are in turn passed to a *lexical tree* matcher that identifies the patterns according to the noun phrase identified at the highest level in the node. For instance the sentence by applying pattern 1 in Table 5.1: NP_1 such as NP_2 to the following sentence:

> *"high and low level programming languages* ***such as*** *Assembly, Java, and C++"*

will extract NP_1 as the entire phrase of *high and low level programming languages* instead of *low level programming languages.* The Stanford token matcher [54] is used to apply the patterns to the tokenized sentence to obtain this result. The extracted candidate sentences are then parsed by identifying the sets of X and Y words comprising each of the hypernym and hyponym phrases, respectively. Stemming and removing initial stem words that are common adjectives is performed to cleanse the phrase prior to the hypernym detection step.

5.3.2 Iterative Semantic Learning

Syntactic extraction generates a set C of candidate pairs that need to be classified as acceptable pairs or ones that should be discarded. The iterative learning steps

```
Input  : Candidate sentences set $C$
Output : Tuple set $P$
Initialize: $P \leftarrow \emptyset$ while $newTuples \neq \emptyset$ do
    $newTuples \leftarrow \emptyset$ foreach $s \in C$ do
        $(X, Y) \leftarrow$ SyntacticExtraction(s)
        if $|X| > 1$ then
            $\chi \leftarrow$ ngrams(X)
            $x_1, x_2 \leftarrow$ maxLikelihood($\chi$)
            $r(x_1, x_2) \leftarrow \frac{p(x_1 \mid Y)}{p(x_2 \mid Y)}$
            if $r(x_1, x_2) > \zeta$ then
                $X \leftarrow x_1$
            else
                $X \leftarrow x_2$
            end
        end
        if $|X| = 1$ then
            $newTuples \leftarrow (X, Y)$
        end
    end
    $P \leftarrow newTuples$
end
return : $P$
```

Algorithm 3: Iterative Semantic Learning Algorithm

populates a set P of valid pairs that is initially empty. A first pass over the sentences will add any pair for which the hypernym phrase contains only a single word. If the hypernym phrase contains more than one word, the phrase must be processed by the hypernym extraction algorithm for phrase-detection. If a valid hypernym phase is detected we add the triple to the set P. We note here that since we used a lexical token matcher, the hyponym phrases required very little processing for validation. Thus, we simply used stemming and lemmatization to reduce them to their base form.

The challenge is extracting a concept and instance in an *is-a* pair is the identification of the concept to a precise level. Instances are typically listed by authors with little descriptors when they are enumerated in a sentence, thus the pattern recovers them in a more clean form than the concept. The concept is usually a noun phrase that can sometimes be a full sentence and more common than not can be reduced to a much narrow noun phrase or single noun. Thus, for our hypernym (concept) extraction algorithm we apply a modified version of the probabilistic approach used in [40]. Given a candidate {X,Y} pair, we detect the values of $x_i \in X$ for the hypernym phrase. This may be a single value of x, or a subset, which we use as a sort set by computing the n-grams from the set X. For that, we compute the likelihood ratio between the two words with highest likelihood of X as follows. For each $x_i \in X$ and $\chi_j \subset X$, where χ_j is a subset of ordered n-gram words from X, we compute:

$$p(x_k \mid Y) = p(x_i)\Pi_{i=1}^{n} p(y_i \mid x_k)$$

where p(x_i) is the percentage of pairs in the P that contain the word x_i in the hypernym phrase, and p($y_i \mid x_k$) is the percentage of pairs in P that contain the term y_j in the hyponym where x_i is in the hypernym. The same likelihood value is computed for each χ_i as well. The highest two likelihood values obtained are then used to compute the ratio r(x_1,x_2):

$$r(x_1, x_2) = \frac{p(x_1 \mid Y)}{p(x_2 \mid Y)}$$

If the ratio is higher than a specified threshold we select x_1 as the hypernym word (or phrase in case it is a χ_1), otherwise x_2 (or χ_2) is selected as the hypernym.

5.4 Scientific Taxonomy Construction

The knowledge base extracted from the previous step in the set P is a set of N hypernym-hyponym pairs. In order to construct a taxonomy graph from these pairs

```
Input  : Local taxonomy graphs set $T$
Output : Taxonomy Graph $G$
$G \leftarrow \emptyset$ foreach $t \in T$ do
    if $|t| = 2$ then
        $G \leftarrow$ node($t_2, t$)
    end
end
foreach $t_i, t_j \in T$ do
    if $t_i.children \cap t_j.children \neq \emptyset$ then
        $G \leftarrow$ merge($t_i.root, t_j.root$)
    end
end
foreach $t_i \in T$ do
    foreach $v \in t_i.children$ do
        foreach $t_j \in T$ do
            if $t_j.children \cap t_i.children \neq \emptyset$ then
                $G \leftarrow$ merge($v, t_j.root$)
            end
        end
    end
end
return : $G$
```

Algorithm 4: Taxonomy Graph Construction Algorithm

we first represent each pair as a local taxonomy tree rooted at the hypernym. The hyponym will be a child node. The construction of the taxonomy is thus reduced to a graph merging problem where initially we have N local taxonomy trees. We define three rules used as a basis for the taxonomy construction.

Axiom 1 (Modifiers). *A noun phrase comprised of a modifier and a noun, is almost always a subconcept of the noun. Thus, if* $(X, Y) \in P$ *and* $X=\{x_1,x_2\}$*, then* $(x_2, X) \in T$.

For example, we can easily infer from the phrase *learning algorithm* that it is a subconcept of the more abstract concept *algorithm.* This allows to identify the specific type of algorithm for other instances in other pairs by adding them under their specific concept, this process is covered by the other merge operations.

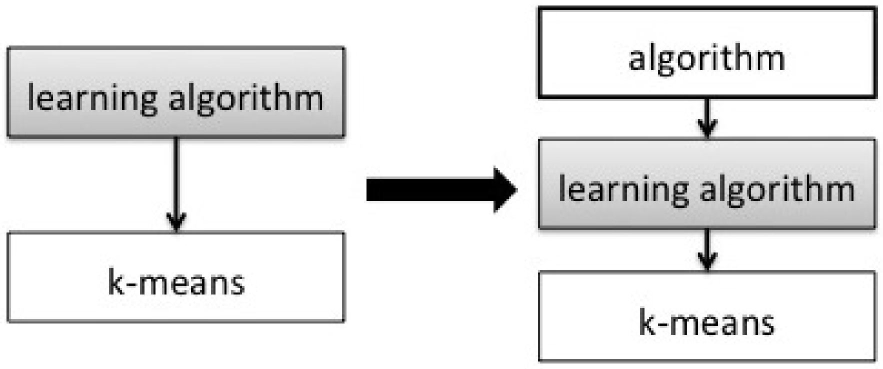
In order to construct the taxonomy from the individual pairs, pairs containing the same concepts must be merged into a single node in the graph. Simple surface form comparing will not do in this case because of possible polysemous terms and phrases. Thus we derive the following two propositions to address the case of several meanings for the same surface form. following proposition:

Axiom 2 (Horizontal Merging). *For two pairs with same text values for their hypernym noun phrases, the noun phrases have the same meaning if they were extracted from the same sentence. If they are not in the same sentence, they have the same meaning if they have similar instances. For pairs* $(X_1, Y_1), (X_2, Y_2) \in P$*, if* $X_1 \cap X_2 = X_1 = X_2$ *and* $Y_1 \cap Y_2 \neq \emptyset$*, then* $(X_1, Y_1 \cup Y_2) \in T$*.*

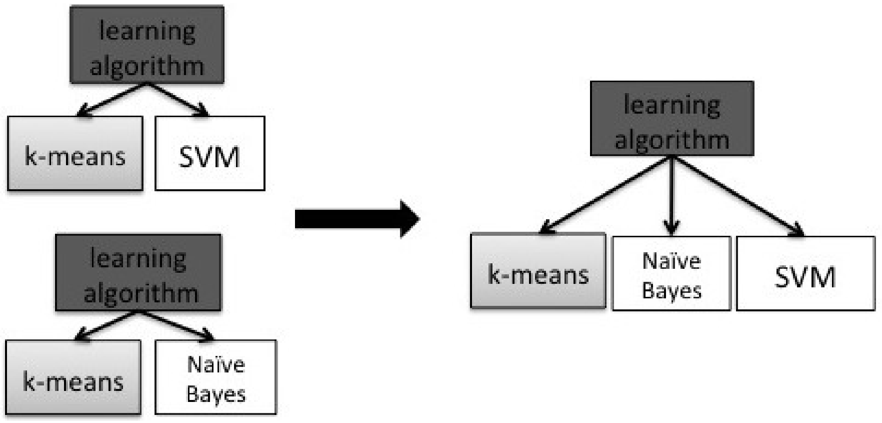
Axiom 3 (Vertical Merging). *For two pairs, if the instance noun phrase of the first pair has the same text value as that of the concept of the other pair, they have the same meaning if the instance's coordinate terms are similar to the instances of the concept in the other pair. For pairs* $(X, Y_1), (V, Y_2) \in P$*, if* $Y_1 \cap V = Y_1 = V$ *and there exist other* Y_i *and* Y_j *such that* $(X_1, Y_i) \in P$*,* $(V, Y_j) \in P$ *and* $Y_i \cap Y_j \neq$ *then* $(X, V) \in T$*.*

Our taxonomy construction algorithm is described in Algorithm 4. Initially the taxonomy graph is empty. Based on the our first proposition, the method performs a `ModifierMerge` operation. This generates a new local taxonomy t of a concept-subconcept relationship. This steps adds these nodes to the set of local taxonomies. This is illustrated in Figure 5.2a.

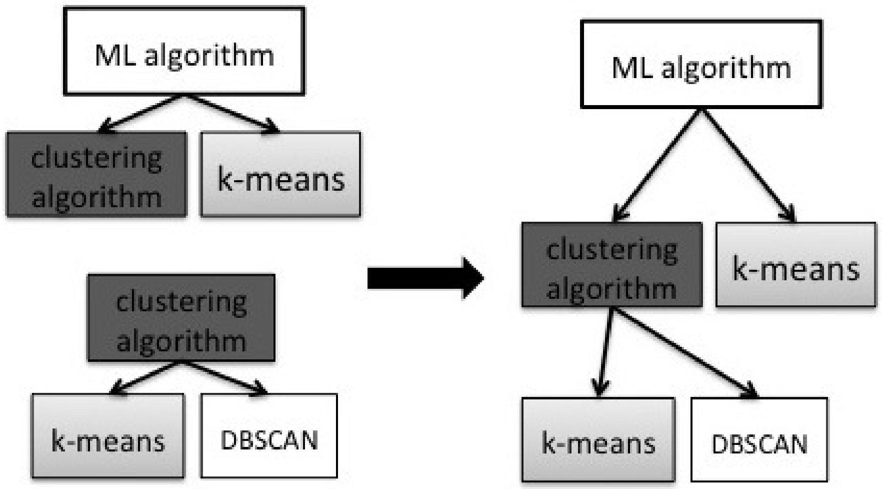
Next, using the second proposition, we perform a `SimpleMerge` operation, where nodes whose roots are similar and belong to the same sentence are horizontally merged into one parent node with the union of children of both nodes. In the case where the two nodes did not occur in the same sentence, the similarity between their children is used to determine whether to merge them or not. If the children have some overlap then we consider the nodes as the same node and merge them into one node. An example is shown in Figure 5.2b



(a)



(b)



(c)

Figure 5.2: (a) Modified noun node insertion. (b) Simple similar root node merge. (c) Vertical similar node insertion.

The third proposition is used for the `VerticalMerge`, shown in Figure 5.2c, which aims to increase the hierarchy level in the graph. We apply this merge by observing the overlap between the each child of a node and the root nodes of other local taxonomies.

Chapter 6 Evaluation of Chart Data Extraction

In this chapter we desrcibe our evaluation for the various components of the bar chart semantic structuring as well as the knowledge base construction methods.

6.1 Dataset

Our entire dataset consists of $40,000$ figures extracted from 10,000 articles published in top fifty computer science conferences between 2004 and 2014. The figures were extracted using a recently released system *pdffigures* [55]. Their system produces an image file and a JSON metadata file for each figure in the document. The metadata contains the location of the bounding box of the figure within the document page and caption. For figures embedded as vector graphics, (eps/ps/PDF), the metadata also contains the text and the bounding box of the words inside the figure. We observed that around 50% of these figures contained sub-figures: often a combination of line graphs, bar charts, and pie charts. By manual inspection of the extracted images we found roughly 1,100 of these documents contained bar charts. By taking a random sample of these charts we found that 67% of them follow the assumptions followed in the data extraction step. To conduct our experiments we

	Precision	Recall	Type-II error
bar extraction	86%	89 %	11%
x-label region	83%	88%	12%
x-label text	73%	79%	21%
x-name char	87%	86%	13%
x-name word	85%	82%	18%
y-name char	77%	74%	26%
y-name word	79%	68%	32%

Table 6.1: Chart-Data Extraction Precision, Recall and Type-II Error

use a subset of 300 bar charts extracted from over 1,000 PDF files.

6.2 Rule-Based Extraction

Data extraction involves correct recognition of a chart's bars, y-axis scale, axes labels and names. In this section we describe our evaluation for the rule-based extraction.

We run experiments on the extracted charts and through manual inspection determine the accuracy of: bar recognition, y-scale calculation, extraction of x-labels and axes names, where they exist. Table 6.1 shows precision, recall, and type-II error for these steps. As noted the recall is higher than precision for the bar extraction, which implies that the connected component method is effective in capturing the bar shaped components in the graphic. Bars that go undetected either have very small height, or are the same color as the background, e.g. white bars in white-background charts. However, lower precision implies that a simple heuristic based approach to classify these bar shapes as data bar or noise, leaves room for improvement.

Y-Scale Extraction The scale of the x-axis, which is the graphic-pixel to real-data ratio, can only be correctly calculated if both the y-axis labels are correctly located and if the OCR returns their correct text values. The locating of y-label regions

-since it is computed as simply the region left to the leftmost bar- depends on the correct identification of the bars. Thus, we evaluate only the accuracy of the y-scale as either correct or incorrect, since it directly relies on a step that has already been evaluated. The method is able to identify 77% of y-scales correctly. This result is lower in charts extracted from PDF documents as opposed to charts from the web or ones from chart design tools [3]. Many interesting factors contribute to this rate, such as the quality of the charts and accuracy of chosen OCR for the chart resolution.

X-Axis labels The x-axis labels are used to name the bars. Since many text strings can appear below the x-axis, we evaluate both the correct identification of the label's location and the correctness of the text. Table 6.1 shows the accuracy of extraction of the label region and the results for correct recognition of the text string it contains. Most cases where the method fails to identify the regions correctly is when the extracted bars contain many false positives. This will cause inaccuracies in the location of the x-axis, and consequently the x-axis labels. In it's current state the method assumes labels are horizontally aligned, thus if a single label spans more than one line, only the string in the first line will be returned. As for the text accuracy, the method does not address slanted labels, although their regions can be identified, the OCR does not recognize the texts. Additionally, if the labels are too small, the text resolution will not be high enough to be recognized.

Axes Names Table 6.1 shows the results for the extraction of the axes names. In many cases, the text characters are correctly extracted however the isotropic dilation may not be able to separate them into their individual words. Thus, we provide a detailed evaluation, one for the character recognition and another for word recognition. Intuitively, the word recognition, i.e. separating the characters into individual words, shows lower performance when the words are placed too close to one another. Sometimes when the image is scaled in one dimension more than the other before it is embedded in the PDF document, the character spacing becomes smaller than an average space for it's font size, which can affect the recognition results. However, the character recognition relies mainly on the OCR's performance. If a word has more than 90% of it's characters returned correctly it is considered correct. The character recognition was noted to perform with better accuracy when the font used is not in bold, with a reasonable image resolution. Certain characters were commonly misinterpreted by the OCR, such as 'r' and 'Q' as well as special characters such as '%' and '#' signs.

6.3 Machine Learning-based Extraction

In this section we describe the evaluation of our method on a set of 213 bar charts extracted from over 1000 PDF files. The tool used to extract the charts is PDFFigures [55], [56]. The charts were selected randomly from the PDF documents, however, to obtain a more diverse set of charts, we reject a chart if the set already contained a chart extracted from that PDF. Most charts from the same document have the same layout. The PDF documents are articles published in top Computer Science conferences.

6.3.1 Graphics Role Labeling

The first part is to evaluate the classification of the graphic components. Tables 6.2 and 6.3 shows the results for multi-class and binary classification using the c4.8 classifier and the random forest classifier with 10-fold cross validation. As shown the random forest produced better results and higher accuracy. It is noted that the bar accuracy is higher and that is due to the fact that bars have more consistent layout in the chart as opposed to legend boxes. Also, the removal of small size components during the connected components labeling step, can cause some legend

Table 6.2: Graphics Components Role Classification Accuracy using Multi-class C4.8 and RF Classifiers

	Precision		Recall		F1	
	RF	C4.8	RF	C4.8	RF	C4.8
bar	98.4	97.5	97.4	96.8	97.9	97.2
legend	92.5	89.9	93.5	88.7	93	89.3
Other	94.7	92.9	96.3	94.3	95.5	93.6

Table 6.3: Graphics Components Role Classification Accuracy using Binary C4.8 and RF Classifiers

	Precision		Recall		F1	
	RF	C4.8	RF	C4.8	RF	C4.8
bar	98.5	97.3	97.2	96.8	97.9	97
legend	93.6	90	90.4	86.3	92	88.1
Other	-	-	-	-	-	-

boxes to be filtered out of the image all together.

6.3.2 Text Role Labeling

The extracted text regions are classified into one of 7 roles and the accuracy is reported for both the C4.8 classifier and random forest. Tables 6.4 and 6.5 shows the precision and recall for each of the roles (the first column) obtained by each of the classifiers applied as multi-class and binary. As noted, the random forest classification scheme provides higher accuracy for the text role labeling. The results are highest for the y-scale values, which is a very important field, since the extraction of the y-scale, and consequently the data values, rely on the correct extraction of these values. The title of the x-axis has lowest accuracy, and that is due to the fact that many charts in our data set did not contain an x-axis title.

Table 6.4: Text Role Classification Accuracy using Multi-class C4.8 and RF Classifiers

	Precision		Recall		F1	
	RF	C4.8	RF	C4.8	RF	C4.8
Y-title	96.8	93.4	95.8	91.6	96.3	92.5
Y-label	96.4	95.8	99.1	98	97.7	96.9
Legend	87.5	81.2	90.9	85.9	89.2	83.5
X-labels	95.1	93.8	95.9	95.1	95.5	94.4
X-title	85.7	84.7	82.4	77.8	84	81.1
Chart-title	88.2	84.7	86.5	83.1	87.4	83.9
Other	91.9	83.5	80.4	74.4	85.8	78.7

Table 6.5: Text Role Classification Accuracy using Binary C4.8 and RF Classifiers

	Precision		Recall		F1	
	RF	C4.8	RF	C4.8	RF	C4.8
Y-title	96.9	94.3	93.8	90.8	95.3	92.6
Y-label	97.4	95.6	98.6	97.6	98	96.6
Legend	93	84.6	85.1	82.8	88.9	83.7
X-labels	96	94.8	95.2	95	95.6	94.9
X-title	89.3	80.1	76.8	76.4	82.6	78.2
Chart-title	92.2	85.4	81.9	80.8	86.8	83
Other	-	-	-	-	-	-

6.4 Feature Extraction

The features extracted from the images are salience, trend, maximum and minimum, time-line and ordinality of the x-axis. Salience refers to when a specific bar has special design features that are meant to draw attention to it. Design features that can be used to express a bar's salience are giving it a different color than the other bars, or by annotating it with a text string. By examining the charts in our data set we found that less than 1% contain these salience features, which is not a sufficient size to evaluate the extraction of this feature.

To evaluate the identification of the trend feature, we first evaluate the extraction of the different bar series and groups. A bar series is the bars that belong to a single data series and are represented graphically by having the same color. A bar group is the collection of bars that have the same x-label. Evaluating these two extracted values is key because in our specific dataset of scientific charts the trends are mostly expressed over a data series or among a group. Since at this point we do not extract the legends of the chart, we determine the chart's trend based only on the trend of each data series, since the summary can be generated by using the x-axis name as the parameter. An appealing extension to this step is to extract the legends and their names so that they can be used as the parameter for the trend protoform, yielding an even more accurate description. Table 6.6 shows the results for the series and group identification steps. It is important to note that this step is based on the correct identification of the y-scale. Charts with a y-scale that was not extracted in the data extraction module, i.e. returned 'nil' by the extractor, cannot proceed to this step, thus the accuracy is calculated over only charts whose y-scale was extracted. Table 6.6 also shows the percentage of correctly identified maximum and minimum values. This is calculated over the entire set of bars in a chart. The fact that the bar extraction overlooks very small bars contributes to a great percentage of incorrect minimum values.

The last feature extracted is determining whether the data is a time series and whether the x-axis is a trend of ordinal values. Table 6.6 shows the percentage of correctly identified time series and ordinal x-axis values. Two factors affect the accuracy of this step: the first is the accuracy of the x-label extraction and correct recognition of their textual values. The second, is that the method in it's current design classifies the x-axis data as time-line only if all the x-axis labels are classified as such. However, the case where only a single label is incorrectly extracted is not uncommon and in the future this occurrence should be tolerated to achieve better detection rates.

Feature	Extraction Accuracy
bar series	100%
bar groups	96%
trend in series	81%
trend in group	79%
x-axis timeline	89%
x-axis ordinal	65%
maximum value	88%
minimum value	68%

Table 6.6: Feature Extraction Evaluation Results

6.5 Bar Chart Data Recovery Accuracy

The quality of the role labeling phase naturally affects the accuracy of the final data values we recover from the chart. To evaluate the effectiveness of our machine-learning approach where we train the classifiers to label the roles of the components, we compare the data extraction accuracy using our methods with those obtained by a rule-based approach proposed in [3]. The data set used for this evaluation is a set of 50 charts different than those used in training our classifier. Tables 6.7 and 6.8 show the precision and recall for the data extraction. As expected the precision achieved using the ML approach is higher for most of the values. Legends had low values for both precision and recall because the evaluation did not tolerate any type of error in the recovery. If the entire legend map was not recovered fully it was considered a miss. Also, in some cases when the legend texts were correctly labeled as being legend text, they were associated with an incorrect legend box, which is considered erroneous in our evaluation. Chart titles have notably high accuracy, this is due to firstly, most charts had the title at the top. This is a general observation about the data set. Additionally, for the data set used in this experiment only 11 out of the 50 charts contained a title. Which is also noted as common in Computer Science papers as not many of the charts contain a title.

Table 6.7: Data Extraction accuracy for Rule-Based vs. Machine Learning Role Labeling -1

	Precision		Recall	
	RB	ML	RB	ML
Data Values	89.65	98.14	93.48	78.06
X-labels	40.03	91.93	84.32	79.26
Legend	-	59.26	-	43.75

Table 6.8: Data Extraction accuracy for Rule-Based vs. Machine Learning Role Labeling -2

	Accuracy %	
	RB	ML
X-title	75.76	90.91
Y-title	80.95	95.24
Y-scale	63.27	78
Chart-title	-	99

6.6 Summary generation

The final step is to evaluate the final summaries, which is an evaluation of the entire system performance. In order to determine the effectiveness of the summaries we evaluate the summaries based on two metrics: fact accuracy and summary relevance. To determine the fact accuracy we evaluate whether the information provided by the summary is simply correct or not. Facts+ indicates an optimistic evaluation, where minor errors that can be corrected by existing tools are overlooked, such as a single incorrectly retrieved character in a word. The Facts- measures the accuracy with a pessimistic evaluation where the summary facts are considered correct only if the data has been extracted with 100% accuracy. The relevance is based on whether the user thinks the summary captures the most important information the chart is designed to illustrate. However, this type of evaluation requires a quantification of the summary relevance in order to be able to measure it. In our evaluation, we assume a summary is considered relevant if it succeeds to mention either trend, maximum and minimum value over a timeline, or a trend in

	Summary Accuracy
Facts+	74%
Facts-	56%
Relevance	83%

Table 6.9: Generated Summary Evaluation

the ordinal values of the x-axis. Or if it successfully concludes the overall trend or max/min values whichever is present, in the case where the chart x-axis is neither a timeline nor contains ordinal value trend.

Since the bar values exhibit a trend of increasing value and the x-axis values are ordinal values, the protoform 2 is applied here. It is noted though, that a large number of charts in scientific documents are used to illustrate experimental results. The results show that most summaries express trends and maximum and minimum values. This is consistent with the fact that not many contain salience features. Table 6.9 shows the result of user evaluation of the summaries generated by the charts. This step also was only evaluated for charts whose y-scale was extracted.

Chapter 7 Evaluation of Knowledge Base Construction

In this chapter we discuss a detailed evaluation on the knowledge base construction method. We evaluate the proposed algorithms as well as discuss the properties of taxonomy that is built using the proposed approach. First we discuss the details of our data set, and then describe the experimental setup used to evaluate the knowledge base construction portion of this thesis.

The first step is to evaluate the quality of the proposed hyponymy extraction method. The second is to evaluate the quality of the taxonomy construction method. Since no ground truth labeled data is available at the time of the study, we adopt the most common evaluation measures used in prominent studies with similar problem [40], [57], [32]. For the hyponymy extraction evaluation we define two evaluation metrics: correctness using precision and recall, and relevance using scoring scale well known in the literature [57]. The taxonomy construction experiments are designed to capture the quality of the resulting knowledge base in terms of: quality of entity categories and entity coverage of the taxonomy.

The hyponymy extraction method was applied to a data set of 10k research articles from CiteSeerX[1] published between 2004 and 2014 in top 50 computer

[1]http://citeseerx.ist.psu.edu

Table 7.1: Comparison of taxonomy graph properties.

	# tuples	Avg. # children	Avg. depth	Max depth
UG-PL	10,589	0.0001	1	3
NG-PL	15,066	0.001	1	3

science conferences. The topics covered include data mining, artificial intelligence, computer security and most computer science topics.

7.1 Hyponymy Extraction

In this section we describe two separate experiments conducted to evaluate correctness of the extracted triples and relevance of the triple statement, respectively.

7.1.1 Correctness

To evaluate the quality of our extracted hypernym-hyponym pairs, we randomly select pairs extracted from 400 documents. We extract the *is-a* pairs from these documents using our approach that incorporates the ngrams of the hypernym phrase and compare the results with that of applying the SuperConceptDetection algorithm described in [40]. We refer to our approach as NG-PL, from n-gram probabilistic learner and the baseline as UG-PL, since it considers only unigrams. From the 400 documents the amount of harvested pairs is 133 and 74 when using our method and the baseline respectively. The triples were annotated by three computer science graduate students. The annotators were presented with the triple and the text containing from which it was extracted and is asked to assign a score to each extracted triple. The scoring system uses a scale ranging 0-4, similar to the one in [57], defined as follows:

0: The extracted relation is nonsense.

1: The extracted relation is either vague or unhelpful. This covers the case where the hypernym-hyponym pair is too abstract to be useful or the phrase is not descriptive enough to substantiate a clear relationship.

2: Opinion/I don't know. An example of this case the pair may contain a very specific concept that cannot be known without knowing the context of the paper it was extracted from, such as a new algorithm name that is just introduced in the paper and the annotator is not familiar with it.

3: The extracted relation is somewhat true. This is used when the pair is generally correct but requires only small modification to become fully true.

4: The extracted relation is correct as is.

The inter-annotator agreement was computed among the three annotations according to the average score given for each pair. The agreement result is 71% and evaluation results are shown in Table 7.3.

7.1.2 Relevance

To evaluate the relevance of the extracted triples we ask the annotators to evaluate the extracted triples based on whether they are concrete or abstract. A concrete relationships is where the hypernym is considered to be one of the closest terms or categories the hyponym can be described with. If the hypernym is a general concept and not considered descriptive enough of the hyponym, it is considered to be abstract. Table 7.2 shows the percentages of both as labeled by the annotators described in section 7.1.1. The last row is a measure of whether the extracted phrase is a complete phrase in itself. This evaluates the phrase detection, if the phrase can hold it's meaning and carry the same meaning even when taken out of the context it is mentioned in, it is considered well-phrased.

Table 7.2: Relevance of Extracted Hyponymy Relationships

Measure	% of Triples
Concrete	82.8
Abstract	17.2
Phraseness	93.39

Table 7.3: Annotator scores for extracted triple quality

Score	#NG-PL	%NG-PL	#UG-PL	%UG-PL
4	72	54.13%	27	36.48%
3	26	19.54%	19	25.67%
2	09	06.76%	07	09.45%
1	15	11.27%	14	18.91%
0	11	08.27%	07	09.45%

7.2 Taxonomy Evaluation

By applying the pattern-based extraction step to the 10k documents, we extract candidate hypernym-hyponym sentences. This generates over 17k candidate sentences with average runtime of 0.3 sec/document. The dependency path parsing step is the most expensive step in this phase. In this section we report on the evaluation of the constructed taxonomy graph and the resulting knowledge base as a source for scientific knowledge.

Table 7.4: Knowledge base coverage of top 3 million queries

# of top queryTerms	#of queries covered
10,000	3504
20,000	4323
30,000	4702
40,000	4915
50,000	5810

Table 7.5: Annotator scores for entity categories in knowledge base

Score	%Entity-Concept pairs
4	61.27%
3	04.27%
2	11.77%
1	06.77%
0	15.93%

7.2.1 Correctness

To evaluate the accuracy of the categories assigned to concepts extracted in our knowledge base, we conduct two evaluations. The first is similar to the one in the previous section where we ask users to annotate the categories of a subset of $1,000$ entities. For this task we ask 6 computer science graduate and senior undergraduate students to score the extractions on the same scale of 0-4. The results are shown in Table 7.5.

The second evaluation of the terms is conducted by a comparison with an existing knowledge base. We compare the assigned hypernym for each entity to those extracted by the Microsoft Concept knowledge base[2]. This is built using the methods proposed in [40] and provides an API to look up entities and retrieve their hypernyms, which is referred to as "class" in the API. By applying our taxonomy construction system to the 17k candidate sentences, we harvest over 15k entities and their hypernym. For each entity if the MS concept graphs returns a hypernym for that entity that matches the one extracted by our system we consider it a match. Out of $15,066$ entities, $9,160$ were found in MS concept graph, and $9,026$ of those entities' hypernyms were found in the classes returned by the API.

[2]https://concept.research.microsoft.com/

7.2.2 Coverage

In order to evaluate the coverage of the We compare the properties of the constructed knowledge graph using our hypernym detection algorithm for ngrams against the same method used only for unigrams, as in [40]. Table 7.1 shows the values for the knowledge base size for each method. To evaluate the quality of the knowledge base we asses the coverage of the entities. To do this we perform a concept-space evaluation similar to that used in [40] to evaluate the coverage of the knowledge base. This measures whether the concepts extracted are ones that are frequently queried, the assumption is if a query term is found in our knowledge base, then it has good coverage. To do this, we compile a set of 3 million query logs submitted to the scholarly search engine CiteSeerX during the years 2015 and 2016. Stop-words are removed from the terms in the queries to find the top unique query terms in the entire set. The 3 million queries contained just over 500,000 unique terms. The most frequent terms were selected and looked up in the knowledge base to assess whether the knowledge base can provide more information about the concept.

Table 7.4 shows the amount of queried topics that were found in the knowledge base. Although the knowledge base covers roughly 5% of the query terms, this percentage does increase with the increase of query terms. Several factors are to be noted here. The query logs are submitted to a search engine that contains papers on most scientific fields not only computer science. Thus, the queries we count are not selected for only computer science related queries, as our document set from which we extract the KB was. Additionally, the query terms, although stop-words were removed, they were not stemmed or lemmatized. Neither did they undergo a cleansing step to remove noisy terms.

In order to evaluate the value of the extracted entities we conduct another experiment to measure the amount of entities in our knowledge base that are queried frequently by users. This is the reverse of the previous test, and implies a slightly different insight. If our entities, which are stemmed and lemmatized, are

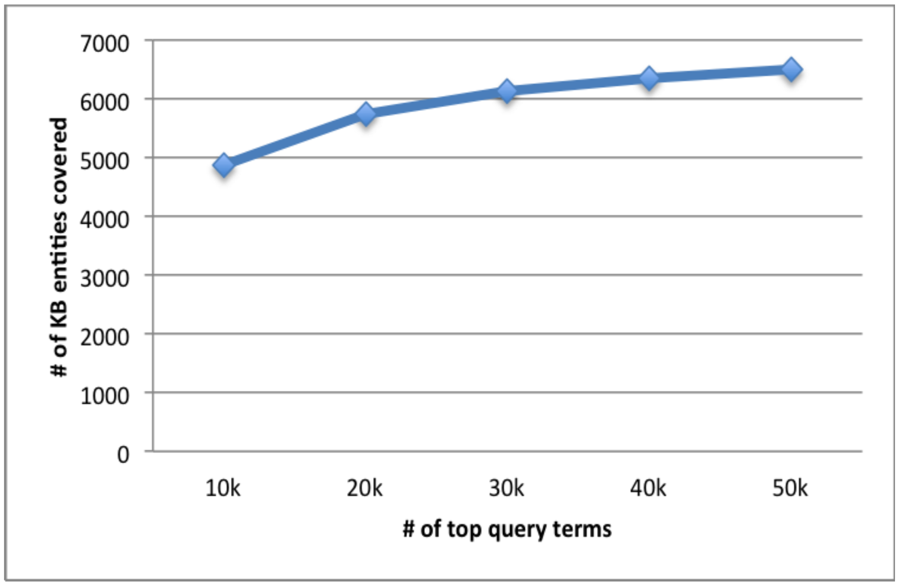


Figure 7.1: KB entities that are frequently queried

queried frequently this suggests that they are useful extractions. Figure 7.1 shows the amount of entities in our knowledge base that were frequently queried. In a set of over 500,000 query terms we count the KB entities that occurred in the top 10,000-50,000 query terms. The results show that 59.76% of the KB entities are among the 0.02% most frequent queried terms, and 79.64% are among the 0.1% of queried terms. About 90% of the KB entities occur at least once in the entire set of 500,000 query terms. This small increase in entity count compared to the large increase in query terms indicates that it is possible that 10% of the KB entities are not useful terms.

7.3 Comparisons with Open Information Extractors

In this section we conduct an experiment to compare the effectiveness of our approach as whole in terms of: the choice of extracting only hyponymy relationships and the quality of the results per the metrics in the previous sections.

Open information extraction (OpenIE) tools such as Reverb [58], [39] and

Table 7.6: Correctness and Relevance of Triples using NG-PL vs. OpenIE Systems on 100 Sentences

System	Phraseness	Correctness	Concrete	Abstract	#Triples
S.OpenIE	149 (38%)	24 (6%)	7 (1.8%)	17 (4.3%)	396
ReVerb	14 (15.9%)	2 (2.3%)	1 (1.1%)	1 (1.1%)	88
NG-PL	16 (69.56%)	13 (56.52%)	9 (39.13%)	4 (17.39%)	23

StanfordOpenIE [59] are designed to extract triples from unstructured text. OpenIE approaches do not assume a predefined schema for the triples extracted, thus the predicate, which is the relationship, can be of any type. In order to evaluate the effectiveness of our approach we compare the results obtained from our system with those obtained by the Reverb and Stanford OpenIE tools.

We run both openIE tools on a set of 100 candidate sentences extracted from over 150 research articles. Both the StanfordOpenIe tool, denoted S.OpenIE, and the Reverb tool extract all relationship types possible, whereas our approach, denoted NG-PL, extracts only hyponymy relationships. Table 7.6 shows the resulting evaluation of the triples extracted by each tool. The first row shows that the StanfordOpenIe tool, denoted S.OpenIE, has the highest recall as it harvests 396 triples from only 100 sentences.The phraseness column reports the amount of triples of which the subject and object were well-formed English phrases. The correctness column measures the triples that scored a 3 or 4 based on our correctness scale described in the previous section. For those correct triples, the quality of the information carried in the triple is measured by whether the fact is concrete or abstract. As noted from the table, our system outperforms openIE tools in quality of phrase extraction, second best is Stanford OpenIE, which is expected since it relies on a dependency parse similar to our approach. Although, the lowest in the amount of extracted triples, our method has the highest ration in quality of triples, i.e. concrete facts.

Table 7.7 shows the amount of triples that are extracted as is-a relationships.

Table 7.7: Hyponymy Extraction using NG-PL vs. OpenIE Systems on 100 Sentences

System	#Triples	is-A	Hyponymy	Phraseness
`S.OpenIE`	396	28	1	17
`ReVerb`	88	8	1	0
`NG-PL`	23	23	13	16

Since both openIE tools extract all types of relationships, the table illustrates the importance of proposing our method for hyponymy extractions as the values show that none of these tools have high precision or recall when it comes to hyponymy relations Extraction. The column displaying hte is-a values, shows the amount of triples that have the is-a as a relationship. However, since "is-a" can be used in English for purposes other than expressing hyponymy, we report the number of these triples that actually are instances of hyponymy use of the is-a predicate. The table shows that our method significantly outperforms both methods in extracting triples if we focus on hyponymy relations, which is the goal of building a knoweldege base.

Chapter 8 Conclusion and Future Work

8.1 Summary and Conclusions

In this paper we propose a summary generation method for scientific charts, specifically bar charts. Our system uses image processing techniques to extract data from the charts and provides an analysis of the data. Charts are given semantic structure by means of semantic-graphs. The summary generation techniques are applied to the labeled graph to produce a textual description of the chart. A main challenge to this work is that the quality of the feature extraction is based on the accuracy of the data extraction method. Data extraction methods have indeed reached high accuracy on this front, however, if a single misread value propagates through the entire pipeline the summary may not be very representative of the chart's content.

Large scale knowledge graph construction from raw text is a challenging problem when the text contains domain specific concepts that are not covered by existing knowledge sources. In this paper we presented a system to extract a knowledge base from scholarly documents to harvest scientific entity-relation pairs. Our approach was tested on a set of 10,000 research articles publish in computer science related conferences. The method bootstraps using a pattern-based approach to extract a seed of triples that are used for learning new ones. An iterative semantic learner harvests more triples by using the knowledge gained from the previous iteration.

We also explore the effectiveness of a probabilistic scheme to identify valid hypernym phrases for extracted triples. A taxonomy graph is then constructed using graph merging and insertion operations designed to provide as much information about entities as possible. Experimental evaluation shows that our hypernym phrase extraction scheme is able to identify 3x more *true* hypernym/hyponym pairs. From a data set of 10,000 documents and over 14,000 candidate sentences, the final taxonomy graph contains over 16,000 concepts. By sampling 450 of those documents we obtain a graph with 113 concepts of which computer science annotators found over 70% of the concepts to be generally true.

8.2 Future Work

The proposed framework is flexible to a myriad of extensions for improving the labeling and summaries. Identifying additional features can result in more complex, and perhaps, more accurate summaries. A notable finding from this study is that salience is not a common message in a data set such as the one used here, i.e. charts found in computer science scholarly documents. This indicates that certain types of messages can be more common in charts in certain contexts. A future extension for this work is to identify the messages that are more pertinent to charts found in computer science scholarly documents.

Future directions for the knowledge base construction problem are three categories: knowledge base construction, knowledge base completion, and knowledge base applications. The first direction can provide more accurate classifiers for the triples. Another future work, may include experimenting with additional bootstrapping approaches to extract more types of relationships other than the hyponymy relationship. The second category is an active area of research, and the knowledge base resulting from this thesis can be used as a foundation for a large academic concept graph that can be further completed and expanded. The third category involves applications built on knowledge base data such as entity based processing of queries as well as question answering problems.

Bibliography

[1] Lu, X., S. Kataria, W. J. Brouwer, J. Z. Wang, P. Mitra, and C. L. Giles (2009) "Automated analysis of images in documents for intelligent document search," *International Journal on Document Analysis and Recognition (IJDAR)*, **12**(2), pp. 65–81.

[2] Liu, Y., K. Bai, P. Mitra, and C. L. Giles (2007) "Tableseer: automatic table metadata extraction and searching in digital libraries," in *Proceedings of the 7th ACM/IEEE-CS joint conference on Digital libraries*, ACM, pp. 91–100.

[3] Al-Zaidy, R. A. and C. L. Giles (2015) "Automatic Extraction of Data from Bar Charts," in *Proceedings of the 8th International Conference on Knowledge Capture*, ACM, p. 30.

[4] Tuarob, S., S. Bhatia, P. Mitra, and C. L. Giles (2016) "AlgorithmSeer: A system for extracting and searching for algorithms in scholarly big data," *IEEE Transactions on Big Data*, **2**(1), pp. 3–17.

[5] Wu, J., J. Killian, H. Yang, K. Williams, S. R. Choudhury, S. Tuarob, C. Caragea, and C. L. Giles (2015) "Pdfmef: A multi-entity knowledge extraction framework for scholarly documents and semantic search," in *Proceedings of the 8th International Conference on Knowledge Capture*, ACM, p. 13.

[6] Al-Zaidy, R. A. and C. L. Giles (2017) "A Machine Learning Approach for Semantic Structuring of Scientific Charts in Scholarly Documents," in *Twenty-Ninth IAAI Conference*.

[7] Nickel, M., K. Murphy, V. Tresp, and E. Gabrilovich (2015) "A review of relational machine learning for knowledge graphs," *arXiv preprint arXiv:1503.00759*.

[8] Weikum, G. and M. Theobald (2010) "From information to knowledge: harvesting entities and relationships from web sources," in *Proceedings of the twenty-ninth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, ACM, pp. 65–76.

[9] Qian, R. (2013), "Introducing the Knowledge Graph: things, not strings," http://blogs.bing.com/search/2013/12/18/bing-in-2013-entities-apps-and-maps.

[10] Singhal, A. (2012), "Introducing the Knowledge Graph: things, not strings," https://googleblog.blogspot.com/2012/05/introducing-knowledge-graph-things-not.html.

[11] High, R. (2012), "The Era of Cognitive Systems: An Inside Look at IBM Watson and How it Works," .

[12] Liu, Y., K. Bai, P. Mitra, and C. L. Giles (2007) "Tablerank: A ranking algorithm for table search and retrieval," in *Proceedings of the National Conference on Artificial Intelligence*, vol. 22, Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, p. 317.

[13] Kataria, S., W. Browuer, P. Mitra, and C. L. Giles (2008) "Automatic Extraction of Data Points and Text Blocks from 2-Dimensional Plots in Digital Documents." in *AAAI*, vol. 8, pp. 1169–1174.

[14] Tuarob, S., S. Bhatia, P. Mitra, and C. L. Giles (2013) "Automatic detection of pseudocodes in scholarly documents using machine learning," in *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*, IEEE, pp. 738–742.

[15] Fang, J., P. Mitra, Z. Tang, and C. L. Giles (2012) "Table Header Detection and Classification." in *AAAI*.

[16] Huang, W. and C. L. Tan (2007) "A system for understanding imaged infographics and its applications," in *Proceedings of the 2007 ACM symposium on Document engineering*, ACM, pp. 9–18.

[17] Yang, L., W. Huang, and C. L. Tan (2006) "Semi-automatic ground truth generation for chart image recognition," in *Document Analysis Systems VII*, Springer, pp. 324–335.

[18] Chen, Z., M. Cafarella, and E. Adar (2015) "DiagramFlyer: A Search Engine for Data-Driven Diagrams," in *Proceedings of the 24th International Conference on World Wide Web Companion*, International World Wide Web Conferences Steering Committee, pp. 183–186.

[19] Chen, S. Z., M. J. Cafarella, and E. Adar (2011) "Searching for statistical diagrams," *Frontiers of Engineering, National Academy of Engineering*, pp. 69–78.

[20] Li, Z., S. Carberry, H. Fang, K. F. McCoy, K. Peterson, and M. Stagitis (2015) "A novel methodology for retrieving infographics utilizing structure and message content," *Data & Knowledge Engineering*, **100**, pp. 191–210.

[21] Elzer, S., S. Carberry, and I. Zukerman (2011) "The automated understanding of simple bar charts," *Artificial Intelligence*, **175**(2), pp. 526–555.

[22] Demir, S., D. Oliver, E. Schwartz, S. Elzer, S. Carberry, and K. F. McCoy (2010) "Interactive SIGHT into information graphics," in *Proceedings of the 2010 International Cross Disciplinary Conference on Web Accessibility (W4A)*, ACM, p. 16.

[23] Savva, M., N. Kong, A. Chhajta, L. Fei-Fei, M. Agrawala, and J. Heer (2011) "Revision: Automated classification, analysis and redesign of chart images," in *Proceedings of the 24th annual ACM symposium on User interface software and technology*, ACM, pp. 393–402.

[24] Chester, D. and S. Elzer (2005) "Getting computers to see information graphics so users do not have to," in *Foundations of Intelligent Systems*, Springer, pp. 660–668.

[25] Chao, H. and J. Fan (2004) "Layout and content extraction for pdf documents," in *International Workshop on Document Analysis Systems*, Springer, pp. 213–224.

[26] Gonzalez, R. and P. Wintz (1977) "Digital image processing," .

[27] Ferres, L., P. Verkhogliad, G. Lindgaard, L. Boucher, A. Chretien, and M. Lachance (2007) "Improving accessibility to statistical graphs: the iGraph-Lite system," in *Proceedings of the 9th international ACM SIGACCESS conference on Computers and accessibility*, ACM, pp. 67–74.

[28] Demir, S., S. Carberry, and K. F. McCoy (2008) "Generating textual summaries of bar charts," in *Proceedings of the Fifth International Natural Language Generation Conference*, Association for Computational Linguistics, pp. 7–15.

[29] Wilbik, A., J. M. Keller, and G. L. Alexander (2011) "Linguistic summarization of sensor data for eldercare," in *Systems, Man, and Cybernetics (SMC), 2011 IEEE International Conference on*, IEEE, pp. 2595–2599.

[30] Zadeh, L. A. (2002) "A prototype-centered approach to adding deduction capability to search engines-the concept of protoform," in *Fuzzy Information Processing Society, 2002. Proceedings. NAFIPS. 2002 Annual Meeting of the North American*, IEEE, pp. 523–525.

[31] Yager, R. R. (1982) "A new approach to the summarization of data," *Information Sciences*, **28**(1), pp. 69–86.

[32] Carlson, A., J. Betteridge, B. Kisiel, B. Settles, E. R. Hruschka Jr, and T. M. Mitchell (2010) "Toward an Architecture for Never-Ending Language Learning." in *AAAI*, vol. 5, p. 3.

[33] Suchanek, F. M., G. Kasneci, and G. Weikum (2007) "Yago: a core of semantic knowledge," in *Proceedings of the 16th international conference on World Wide Web*, ACM, pp. 697–706.

[34] Auer, S., C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives (2007) "Dbpedia: A nucleus for a web of open data," *The semantic web*, pp. 722–735.

[35] Lenat, D. B. and R. V. Guha (1989) *Building large knowledge-based systems; representation and inference in the Cyc project*, Addison-Wesley Longman Publishing Co., Inc.

[36] Bollacker, K., C. Evans, P. Paritosh, T. Sturge, and J. Taylor (2008) "Freebase: a collaboratively created graph database for structuring human knowledge," in *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, AcM, pp. 1247–1250.

[37] Vrandečić, D. and M. Krötzsch (2014) "Wikidata: a free collaborative knowledgebase," *Communications of the ACM*, **57**(10), pp. 78–85.

[38] Etzioni, O., M. Cafarella, D. Downey, S. Kok, A.-M. Popescu, T. Shaked, S. Soderland, D. S. Weld, and A. Yates (2004) "Web-scale information extraction in knowitall:(preliminary results)," in *Proceedings of the 13th international conference on World Wide Web*, ACM, pp. 100–110.

[39] Fader, A., S. Soderland, and O. Etzioni (2011) "Identifying relations for open information extraction," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, pp. 1535–1545.

[40] Wu, W., H. Li, H. Wang, and K. Q. Zhu (2012) "Probase: A probabilistic taxonomy for text understanding," in *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, ACM, pp. 481–492.

[41] Hearst, M. A. (1992) "Automatic acquisition of hyponyms from large text corpora," in *Proceedings of the 14th conference on Computational linguistics-Volume 2*, Association for Computational Linguistics, pp. 539–545.

[42] Cederberg, S. and D. Widdows (2003) "Using LSA and noun coordination information to improve the precision and recall of automatic hyponymy extraction," in *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, Association for Computational Linguistics, pp. 111–118.

[43] Snow, R., D. Jurafsky, A. Y. Ng, et al. (2004) "Learning syntactic patterns for automatic hypernym discovery." in *NIPS*, vol. 17, pp. 1297–1304.

[44] Yates, A., M. Cafarella, M. Banko, O. Etzioni, M. Broadhead, and S. Soderland (2007) "Textrunner: open information extraction on the web," in *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, Association for Computational Linguistics, pp. 25–26.

[45] Cimiano, P., A. Pivk, L. Schmidt-Thieme, and S. Staab (2005) "Learning taxonomic relations from heterogeneous sources of evidence," in *Ontology Learning from Text: Methods, evaluation and applications*.

[46] Miller, G. A. (1995) "WordNet: a lexical database for English," *Communications of the ACM*, **38**(11), pp. 39–41.

[47] Ponzetto, S. P. and M. Strube (2007) "Deriving a large scale taxonomy from Wikipedia," in *AAAI*, vol. 7, pp. 1440–1445.

[48] Swoboda, T., M. Hemmje, M. Dascalu, and S. Trausan-Matu (2016) "Combining Taxonomies using Word2vec," in *Proceedings of the 2016 ACM DocEng Symposium on Document Engineering*, ACM, pp. 131–134.

[49] Fletcher, L. A. and R. Kasturi (1988) "A robust algorithm for text string separation from mixed text/graphics images," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, **10**(6), pp. 910–918.

[50] Vassilieva, N. and Y. Fomina (2013) "Text detection in chart images," *Pattern Recognition and Image Analysis*, **23**(1), pp. 139–144.
URL http://dx.doi.org/10.1134/S1054661813010112

[51] Zhang, X., C. Yang, F. Liu, Y. Liu, and Y. Lu (2014) "Optimizing and scaling HPCG on Tianhe-2: early experience," in *International Conference on Algorithms and Architectures for Parallel Processing*, Springer, pp. 28–41.

[52] Lopez, P. (2008-2016), "GROBID," https://github.com/kermitt2/grobid.

[53] Manning, C. D., M. Surdeanu, J. Bauer, J. R. Finkel, S. Bethard, and D. McClosky (2014) "The stanford corenlp natural language processing toolkit." in *ACL (System Demonstrations)*, pp. 55–60.

[54] Chang, A. X. and C. D. Manning (2014) "TokensRegex: Defining cascaded regular expressions over tokens," *Tech. Rep. CSTR 2014-02*.

[55] Clark, C. and S. Divvala (2015) "Looking Beyond Text: Extracting Figures, Tables, and Captions from Computer Science Paper," in *AAAI Workshop on Scholarly Big Data*.

[56] Clark, C. and S. Divvalla (2016) "PDFFigures 2.0: Mining figures from research papers," in *IEEE/ACM Joint Conference on Digital Libraries (JCDL)*, IEEE, pp. 143–152.

[57] Li, X., A. Taheri, L. Tu, and K. Gimpel (2016) "Commonsense knowledge base completion," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL), Berlin, Germany, August. Association for Computational Linguistics*.

[58] Etzioni, O., A. Fader, J. Christensen, S. Soderland, et al. (2011) "Open information extraction: The second generation," in *Twenty-Second International Joint Conference on Artificial Intelligence*.

[59] Angeli, G., M. J. Premkumar, and C. D. Manning (2015) "Leveraging linguistic structure for open domain information extraction," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL 2015)*.

Vita

Rabah Al-Zaidy

Education

- Ph.D. Computer Science and Engineering, Pennsylvania State University, 2017.
- M.A.Sc. Information Systems Engineering, Concordia University, Montreal, Canada, 2011.

Publications

Journal Articles

- R Al-Zaidy, BCM Fung, AM Youssef, F Fortin. Mining criminal networks from unstructured text documents. In *Digital Investigation (DIIN)* 8 (3), 147-160, 2012.
- R Al-Zaidy, A Kircanski, AM Youssef. Cryptanalysis of the parameterized improved fast encryption algorithm for multimedia/ IEEE Communications Letters 12 (12), 2008.

Proceedings

- Rabah A. Al-Zaidy, C. Lee Giles. A Machine Learning Approach for Semantic Structuring of Scientific Charts in Scholarly Documents. In *AAAI*, 2017.
- Rabah A. Al-Zaidy, Sagnik Ray Choudhury, C. Lee Giles. Automatic Summary Generation for Scientific Data Charts. In *AAAI*-SBD Workshop, 2016.
- Rabah A. Al-Zaidy, C. Lee Giles. Automatic Extraction of Data from Bar Charts. In *Proceedings of the 8th K-CAP Conference.*, 2015.
- R Al-Zaidy, B Fung, AM Youssef. Towards discovering criminal communities from textual data. In *Proceedings of the 2011 ACM Symposium on Applied Computing, 172-177*, 2011.
- A Kircanski, R Al-Zaidy, AM Youssef. A new distinguishing and key recovery attack on NGG stream cipher. Cryptography and Communications 1 (2), 269-282, 2009.