EXAMINING INFORMATION-GRAPHICS: EXTRACTING IMPORTANT INFORMATION FROM USER-WRITTEN QUERIES TO HELP DEVELOP AN EFFECTIVE RETRIEVAL SYSTEM

by

Matthew Stagitis

A thesis submitted to the Faculty of the University of Delaware in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer and Information Science with Distinction

Spring 2013

EXAMINING INFORMATION-GRAPHICS: EXTRACTING IMPORTANT INFORMATION FROM USER-WRITTEN QUERIES TO HELP DEVELOP AN EFFECTIVE RETRIEVAL SYSTEM

by

Matthew Stagitis

I certify that I have read this thesis and that in my opinion it meets the academic and professional standard required by the University as a thesis for the degree of Bachelor of Science.

Signed: ______________________________
M. Sandra Carberry, Ph.D.
Professor in charge of thesis

Approved: ______________________________
Kathleen F. McCoy, Ph.D.
Committee member from the Department of Computer and Information Sciences

Approved: ______________________________
James L. Glancey, Ph.D.
Committee member from the Board of Senior Thesis Readers

Approved: ______________________________
Michelle Provost-Craig, Ph.D.
Chair of the University Committee on Student and Faculty Honors

ACKNOWLEDGMENTS

I would like to thank Professor Sandra Carberry and Professor Kathleen McCoy for overseeing and guiding my research during each of the past three years. They were each helpful in pushing me to overcome any and all obstacles that were faced as the research progressed.

I would also like to thank University of Delaware graduate student Zhuo Li for working with me and helping me to develop my information extraction system. Without her work, it would have been much more difficult for me to complete my thesis within the time allotted.

Lastly, I would like to thank all of my co-researchers at the University of Delaware's Tea House and Millersville University. Together with their work, I feel as though this research will result in significant findings.

This work was supported by the National Science Foundation under grant III-1016916.

TABLE OF CONTENTS

LIST OF TABLES vi
LIST OF FIGURES vii
ABSTRACT viii

Chapter

1 INTRODUCTION 1

1.1 Graph Retrieval 2
1.2 Common Image Retrieval Methods 4
1.3 My Research Problem 5
1.4 Overall Approach 9

2 BACKGROUND 11

2.1 Related Work 11
2.2 Decision Trees 13

3 DATA COLLECTION AND STORAGE 20

3.1 The First Human Subject Experiment 20
3.2 The Second Human Subject Experiment 22
3.3 The Database 23
3.4 Data Correction 24

4 BUILDING THE INDEPENDENT AXIS AND DEPENDENT AXIS MODEL 26

4.1 Global and Local Attributes 26
4.2 Extracting Candidate Entities 27
4.3 The Attribute Set 28
4.4 Annotating the Query-Entity Pairs 29
4.5 Building the Decision Tree 29

5 EVALUATING THE INDEPENDENT AXIS AND DEPENDENT AXIS MODEL . . . 33

5.1 Decision Tree Evaluation Method . . . 33
5.2 Baselines . . . 34
5.3 Results . . . 34

6 BUILDING THE INTENDED MESSAGE MODEL . . . 37

6.1 The Intended Message Categories . . . 39
6.2 Using the Independent/Dependent Axis Content . . . 39
6.3 The Attribute Set . . . 41
6.4 Building the Decision Tree . . . 42

7 EVALUATING THE MODEL FOR HYPOTHESIZING REQUISITE INTENDED MESSAGE . . . 43

7.1 Results . . . 43

8 CONCLUSION AND FUTURE WORK . . . 46

8.1 Conclusion . . . 46
8.2 Future Work . . . 47

BIBLIOGRAPHY . . . 48

LIST OF TABLES

2.1 The dataset for the decision tree example[1] 15

4.1 The attributes for the independent and dependent axes decision tree 30

4.2 Continuation of Table 4.1 31

5.1 The confusion matrix for the axis decision tree 34

6.1 The intended message categories 40

6.2 The attributes for the intended message decision tree 41

7.1 The confusion matrix for the intended message category decision tree 44

7.2 The precision and recall values for each of the intended message categories 45

LIST OF FIGURES

1.1 Sample information-graphics taken from various sources 2

1.2 An information-graphic comparing the annual revenue generated per employee at various technology companies 3

1.3 An information-graphic that would be a great result for query Q_2 . 6

1.4 An information-graphic that would be a great result for query Q_2 . 7

2.1 A sample decision tree for predicting whether or not to play tennis[2] 14

2.2 The general decision tree learning algorithm[3] 16

3.1 A sample information-graphic from the first experiment 21

3.2 A sample information-graphic from the second experiment 22

4.1 A sample query's parsing 27

6.1 A sample information-graphic showing the total number of credit cards in circulation for various companies 38

8.1 The overall method for the information extraction system 46

ABSTRACT

Information-graphics (non-pictorial graphics such as bar charts and line graphs that depict attributes of entities and relations among entities) have attracted increased attention from both the research and industrial worlds. While a lot of attention has been given in information retrieval research to retrieving textual documents, relatively little work has been done to retrieve information-graphics in response to user-written queries. Common image retrieval techniques, such as Image Meta search and Content-based Image Retrieval (CBIR), work well for ranking ordinary images and pictures but often fall short when dealing with information-graphics since they do not take into account their structure and content. This research attempts to analyze a user's query to identify the content of the independent and dependent axes of graphs that might be relevant to the query and to identify the kind of high-level message that would be conveyed by relevant graphs. Natural language processing techniques are used to extract features from the query and machine learning is used to build a model for hypothesizing the content of the axes and the intended message. Results have shown that the constructed models can achieve an accuracy of about 81%.

Chapter 1
INTRODUCTION

Google Search, Bing, and Yahoo Search are three extraordinarily popular search engines that are utilized by internet users who seek information from the World Wide Web. In using a search engine, the user formulates a query which the engine uses to try to retrieve text documents, images or any mixture of the two that fulfills the user's needs. Common search engines have mastered the art of breaking down user-written queries and returning relevant results in the form of online text documents. Typically, such results are found by locating documents that contain the same words that are used within the query itself. Attempting to accurately retrieve non-textual results, however, is much more difficult due to the fact that each potential result's contents cannot be accessed quite as easily. The reason for these difficulties is that the semantics and overall content of an image are more difficult to determine than the plain text found in textual documents.

There are certain tricks and schemes that can be implemented to achieve better results when performing image retrieval. For example, the relevance of an image can be evaluated by matching a query's words to the words contained in the paragraph of the resident document that is closest to the image itself. This strategy, however, suffers when the paragraph that is most relevant to the graphic does not appear adjacent to the image within the document or when the text within the document fails to mention information portrayed by the image [3]. In addition to word matching, low-level features such as the colors contained within the image may also be used to evaluate a potential result's relevance. For example, color is a useful characteristic for a retrieval system that takes an image as input and attempts to return similar images [17].

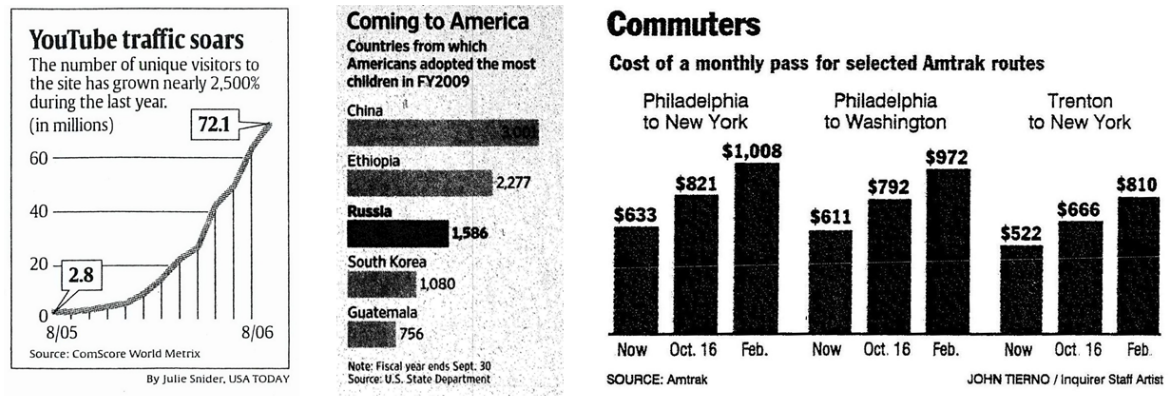


Figure 1.1: Sample information-graphics taken from various sources

However, the most successful systems still suffer from weaknesses, one of which is the retrieval of information-graphics.

1.1 Graph Retrieval

Information-graphics are non-pictorial graphics that depict attributes of entities and relations among entities. They are often displayed as line graphs, bar charts, or pie charts in everyday magazines and newspapers. Figure 1.1 shows some examples of the kinds of graphics we are interested in retrieving. An author might wish to retrieve such an information-graphic in order to include it in a story they are writing or in order to understand relationships between the entities depicted.

Consider, for example, an author who is constructing a report and wishes to make a point concerning the annual revenue generated per employee at top technology companies. First, the author has the option to describe in words how the companies rank amongst one another based on this statistic. On the other hand, imagine that the author has the ability to query a system that allows him or her to input a specific query and retrieve an information-graphic that illustrates information that could be used to answer the query. The following query, which will be referred to as query Q_1, might be an example of such a query:

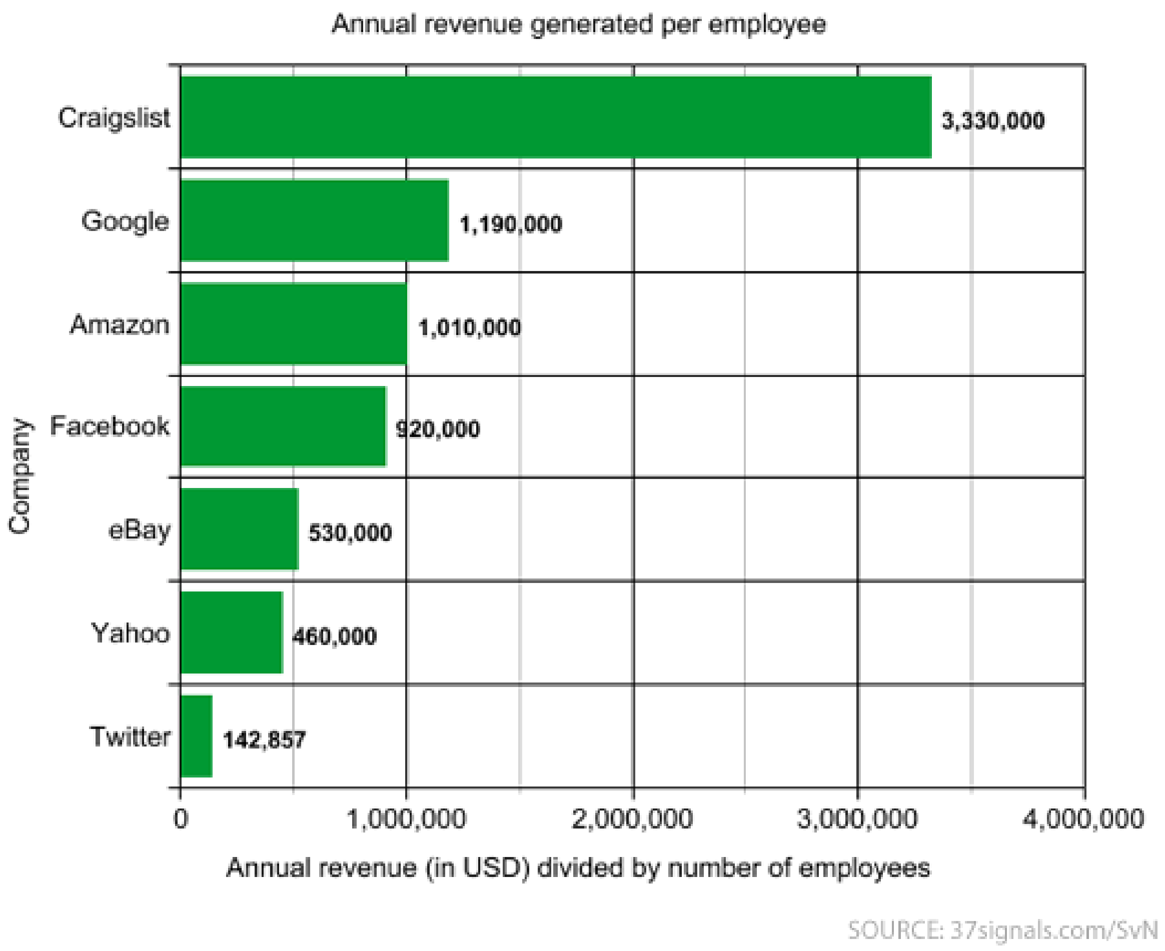


Figure 1.2: An information-graphic comparing the annual revenue generated per employee at various technology companies

> Q_1: *How does the amount of revenue collected per employee compare amongst large technology companies?*

Figure 1.2 shows a simple information-graphic that would be an ideal response to query Q_1. The author's point is immediately made much more powerful and convincing once he or she includes this graphic within the article. Unfortunately, most commercial search engines are unable to respond effectively to requests for information-graphics. This is largely due to their inability to understand the content of the graphic. Instead, the search engines may rely on the text of the document containing the graphic while paying special attention to the image tag information and image file name. For example, on 3/6/2013 query Q_1 was input into a Google Image Search. The first graphic that was returned compares the annual compensation per paid employee for private education establishments amongst industry groups in the Philippines. Likewise, the majority of the information-graphic results were similarly off-target due to the retrieval system's reliance on the existence of the query's words in the text of the webpage source file. By relying heavily on this heuristic, the search engine fails to take the graphic's content into consideration.

1.2 Common Image Retrieval Methods

Search engines currently treat information-graphics and ordinary pictures very similarly. Most modern search engines use one of two different methods when deciding a potential result's relevance: Image Meta search and Content-based Image Retrieval (CBIR) [5]. In the Image Meta search technique, each potential result's metadata is stored in an extremely large database [2]. This metadata is normally a combination of keywords, text within the graphic and text found around the graphic in its resident document. Once a user submits a query, the search engine will then compare said query to the metadata at each indexed entry in the database; thus images will be returned whose metadata has words that are contained in the query. The greater the overlap in words, the higher the graphic will be ranked. The images are returned to the user in a specific order as decided by the search engine's ranking algorithm.

On the other hand, Content-based Image Retrieval applies computer vision techniques during the retrieval process to use aspects of an image's visual appearance rather than solely relying on text [14]. Features such as colors, textures and shapes are often extracted by systems implementing CBIR and then used when evaluating results. If a user is searching for an image of a house, it is possible that the vision system knows the characteristics of houses and can use them to locate relevant results. The drawback of a CBIR system attempting to return information-graphics is the fact that it does not attempt to extract the information that the graphic displays but only considers its physical characteristics and appearance. Unlike ordinary images that consist of only low-level features, information-graphics are given a structure by the independent and dependent axes that allows for them to relay additional information in the form of relationships between the two axes. For instance, information-graphics contain words in specific locations in order to indicate high-level meanings. As a result, both Image Meta search and Content-based Image Retrieval techniques might be able to identify a line graph or bar chart, but they would fall short in that they lack the ability to extract high-level content, such as a potential result's domain or relationships within that domain, when performing a search. Suppose the query, referred to as query Q_2, is as follows:

> Q_2: *What is the percent change in the U.S. GDP, by quarter, from 2005 to 2009?*

If a user was to enter query Q_2 into a Google Image Search, he or she would expect to have an information-graphic, such as the one shown in Figure 1.3, returned as a result. However, this is not the case. Instead, in a search performed on 3/12/2013, the top two information-graphic results were concerned with real GDP growth over the last 30 years and the fluctuating GDP in Japan during the 1980s and 1990s. We argue that for great improvement in retrieval of information-graphics, the retrieval system must consider the specific information need expressed in the user's query and develop methods for retrieving graphics based on that need. In this work, we are concerned with the first step toward that end: how can search engines extract information concerning the content of a desired information-graphic from a user-written query so that it may be used to accurately rank potential results amongst one another?

1.3 My Research Problem

Here we argue that accurate retrieval of information-graphics requires analysis well beyond the vision techniques that rely on low-level visual characteristics, which may be very effective for retrieving pictures. When performing a search of any kind, the best results can be found by evaluating the relevance between the desired pieces of

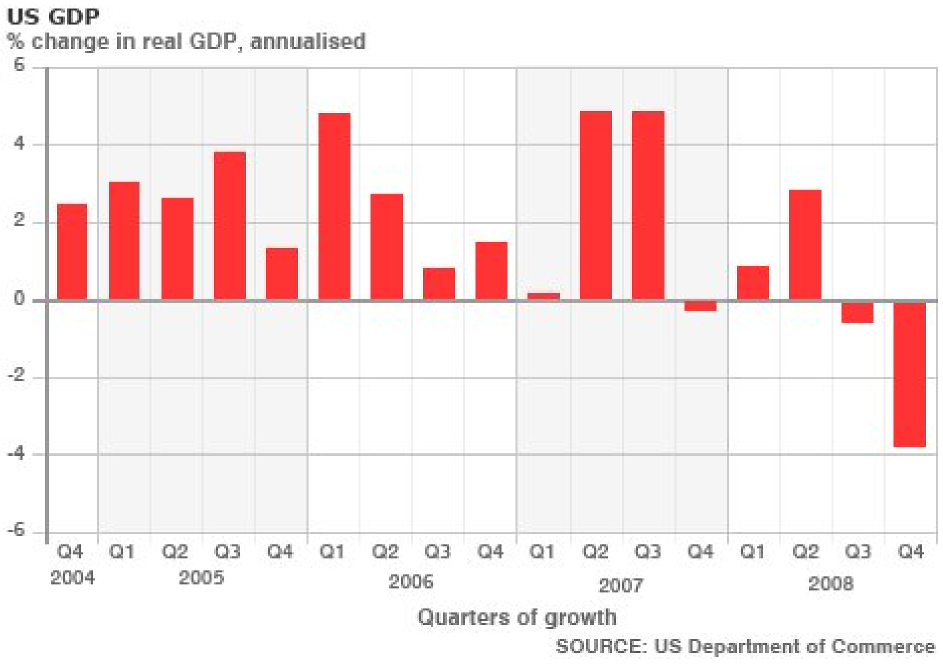


Figure 1.3: An information-graphic that would be a great result for query Q_2

information and the content and characteristics of each possible result. Information-graphics contain captions, independent and dependent axes, and annotations, all of which convey meaning.

This information is useful in conveying what information is captured by the dependent and independent axes, which are important determiners of an information-graphic's content. In this work, we are concerned with extracting the desired content of the independent and dependent axes from a user's query so it can be used to match against graphics during retrieval. Additional work [6] [27] has argued that an information-graphic contains a high-level message that the author intends for the reader to recognize. Essentially, this intended message refers to the high-level message that the graphic was designed to convey. For example, consider the slightly altered graphic that was taken from the web in Figure 1.4. Its intended message is ostensibly that Life cereal ranked highest in terms of votes for favorite cereal as compared to the

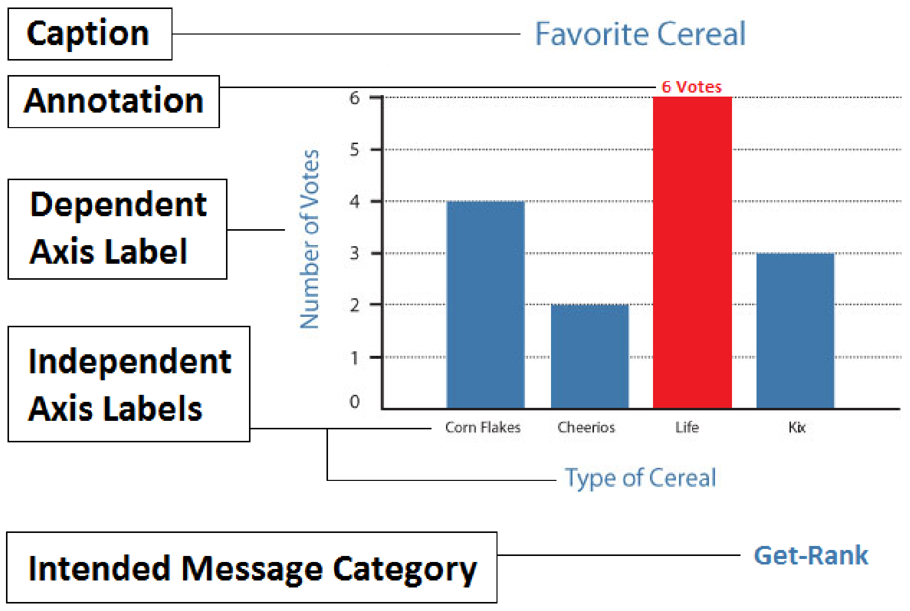


Figure 1.4: An information-graphic that would be a great result for query Q_2

other cereals depicted in the graph. On the other hand, an information-graphic with a trend intended message would presumably have been designed to convey a changing value over an arbitrary period of time. Stephanie Elzer's research describes an exhaustive list of possible intended message categories (such as rank or trend) for simple bar charts [6]; the same may be found for single line graphs in Peng Wu's work [27]. We contend that the intended message of an information-graphic is important to consider in their retrieval and that the desired intended message is often conveyed in a user's query. Therefore, extracting the intended message from any given query is an important component of this work.

The importance of extracting the desired independent and dependent axes from a user's query can be seen in the following examples. Suppose a user is searching for a graphic displaying the numbers of endangered species that live on each of the seven continents using query Q_3.

Q_3: *How many endangered species are found on each continent of the* *world?*

He or she would not be satisfied with an information-graphic that showed how many different continents each endangered species inhabited. However, this information-graphic would be relevant to query Q_4.

Q_4: Which endangered species are found on the most continents?

In the case of these two queries, the desired graphics would be very similar but their independent and dependent axes would be switched. Query Q_3 desires a result with the seven continents of the world listed on the independent axis while the dependent axis measured the number of endangered species found. On the other hand, query Q_4 desires a graphic that lists a finite set of endangered species on the independent axis and measures the number of continents on the dependent axis. Therefore, it is important to analyze a query and identify what should be on the independent and dependent axes of relevant graphs.

Much like an information-graphic's axis content, identifying the appropriate intended message can also improve a retrieval system's accuracy when performing searches. Users, such as the author of query Q_1, might want an information-graphic that conveys the ranking of technology companies in terms of revenue per employee as opposed to a graphic that conveys the trend in revenue per employee at technology companies. Consequently, it is helpful to analyze a query and identify the kind of intended message that will be conveyed by relevant graphs and use this when ranking potential graphs for retrieval.

At this point, it should be clear that to implement an accurate retrieval system, the query must be closely examined to match the content requested by the query with the content conveyed by an information-graphic. More specifically, this study focuses on the extraction of the desired independent axis content, dependent axis content, dependent axis entities, and category of intended message from user-written queries.

1.4 Overall Approach

This research is a first step toward a retrieval system that takes into account how well potential results match the desired axis content and intended message category expressed in a user query. The study will use machine learning to construct decision trees that will extract the previously mentioned content from full sentence queries. The queries must be full sentence queries because the retrieval system will be attempting to analyze the query and extract the content of the independent and dependent axes of relevant graphs. In a search consisting of merely a set of words, also known as a keyword search, it is much more difficult to uncover the general structure of relevant graphics. Suppose that a user has submitted query Q_5 to the retrieval system.

Q_5: *Google Facebook revenue*

From this set of words, one cannot identify the structure of relevant information-graphics. As a result, the retrieval system is unable to decide whether the user desires a graphic showing a comparison of the two companies' revenues, a graphic conveying the trend of the sum of their revenues over a period of time, or a graphic having any other possible structure formed from the three words. Full-sentence queries are required to ensure that the retrieval system is able to use the structure of the query to uncover the structure of relevant graphs. Another key assumption of this research is that users are only searching for simple bar charts and single line graphs. Future research will extend this work to other kinds of information-graphics.

Chapter 2 discusses related work, including how a decision tree is created and used to predict characteristics of user-written queries. Chapter 3 will then describe the data collection and storage methods that were used in this research. Afterwards, Chapters 4 through 7 will walk through the creation and evaluation of the decision trees. In the case of the independent and dependent axis decision tree, each instance will be classified as one of the following: 'X-Helpful', 'Y-Helpful', or 'Not-Helpful'. This value corresponds to which axis (independent, dependent, or neither) the instance under review belongs to. The intended message decision tree will output the category of intended message requested by the query under review. The accuracy of each tree will be evaluated using a cross-validation technique that will be explained in greater detail in the next chapter. Finally, Chapter 8 will discuss possible future work.

Chapter 2
BACKGROUND

Before continuing with the study, it is important to introduce related work along with a few topics that will be used throughout.

2.1 Related Work

Much information retrieval work has been concerned with the retrieval of documents relevant to a user's query. In this work, a document and query are seen as a collection of words. In order to perform document retrieval, a search engine must perform the following general steps:

1) Transform each potential result into a model that can be understood by the retrieval system.

2) Separate out and rank the relevant results in response to a user-written query. It is how each of these steps is implemented that makes any given search engine different from others.

A vector space model is one modern idea "used to model a text document and a user query in an information retrieval system" [1, p. 2]. A vector space model represents retrieval objects as elements of a vector space using algebraic models. "When a query is presented, the system formulates the query vector and matches it against the document vectors based on a chosen method of determining similarity" [19]. Normally, both document and query vectors are created by listing the words contained in them and providing a weight that corresponds to that word's importance, often represented as its frequency. Other common models include extended boolean models [13], latent semantic indexing [22] and probabilistic relevance models [20]. Once a modeling structure has been chosen, document preprocessing is often used to improve a system's performance by carrying out maintenance tasks that help achieve higher accuracies. Examples of such tasks might include the removing of a document's "stop" words or the implementation of case-folding strategies [12]. Case-folding is the conversion of letters between uppercase and lowercase characters that can make string comparison methods simpler. Normally, case-folding is employed by converting all characters to either uppercase or lowercase using the ASCII character set [10].

Once each potential document has been converted into an understandable model, they must each be ranked against the user's query. The ranking algorithm is the most crucial component of a retrieval system because it decides the final ordering of results to be returned to the user. When representing a textual document, each word that appears can be given a value of importance based on the number of times the word appears after the text is tokenized [16]. Typically term frequency-inverse document frequency, or TF-IDF, is used.

TF-IDF combines the term frequency with a measure of how many documents a word occurs in, weighing more heavily terms that occur in fewer documents. Thus, TF-IDF is a method that produces a score that is "more specifically called weight and represents how much a given term has an impact and reflects strongly the meaning of the document" [1]. Raw term frequency is a poor value to use for a term's weight because it considers each term to be of equal importance, even if that term is not discriminative of the document's meaning. Instead, TF-IDF rates more highly words that are unique to a particular document [26]. Vector space models are often used along with TF-IDF weighting to form a solid search model foundation that can be expanded upon with additional techniques [8].

Another successful example of a ranking algorithm is PageRank, "a web ranking technique that has been a fundamental ingredient in the development and success of the Google search engine" [9]. PageRank is based on the assumption that a web page is important if it is pointed to by other important pages. Within the algorithm, each web page is given a numeric value based on the number of pages linking to it, as well as the number of outgoing links and PageRank of those same pages. That numeric value is then used to return the web pages sorted in descending order based on overall importance.

Unfortunately, the algorithms and techniques used in the retrieval of textual documents will not work for my research due to the scarcity of words in information-graphics and the failure of these methods to account for the content reflected by the structure of a graphic. Since this research deals with the retrieval of information-graphics, image retrieval methods are more likely to be useful. As illustrated before, retrieving information-graphics using only relevant text, such as a graph's caption and accompanying text, is normally insufficient. Content-based Image Retrieval has been making significant strides in learning to identify entities and sceneries from natural images. One relevant branch of research attempts to retrieve images based on human sketches, which are then compared to potential results that are ranked according to their differentiation from the user's sketch [17]. The majority of work, however, deals with image retrieval based on color, texture and shape features. Unfortunately, these techniques would not be very useful for my research because they do not address the informational content of information-graphics.

The work presented in this paper is the first real work that places maximum importance on extracting the information that would be presented by the ideal graphic in response to a user's query. In doing so, the retrieval system will be able to exploit knowledge of this information to rank potential results in a more effective manner.

2.2 Decision Trees

A decision tree is a machine learning tool that is used to predict the value of a specific attribute for any instance given the values of the instance's other attributes. The attribute that is being predicted is often referred to as the class attribute. For example, consider Figure 2.1 which shows a simple decision tree that helps decide whether or not a person should play tennis based on the weather outlook, humidity and wind [18]. A single instance in the dataset consists of actual values for each of these three characteristics, also known as attributes. The decision tree can be used to

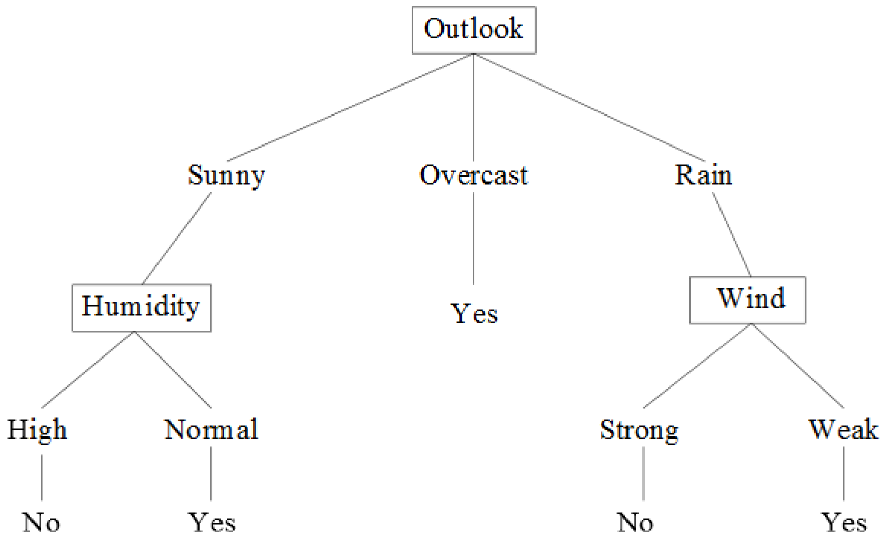


Figure 2.1: A sample decision tree for predicting whether or not to play tennis[1]

decide whether or not to play tennis depending on the values of these characteristics. For example, if the weather's outlook is rain, the humidity is high, and the wind is strong, then the tree indicates not to play tennis.

When traversing a decision tree to classify an instance, always begin at the root, or the top. The root node of the decision tree in Figure 2.1 is marked by the outlook attribute. Since the tennis player knows that the outlook is rain, he or she may slide down the rightmost path from the top of the tree - that is, he or she follows the path corresponding to the value of the attribute being tested. The next node on this path in the decision tree is labeled by the wind attribute. This value is also known by the tennis player so he or she may now slide down the left path denoted by strong wind. Finally, there are no more decision nodes and the tree makes the decision that the tennis player should not play tennis on that day. Overall, such trees serve a practical purpose in choosing a class value based on other observations.

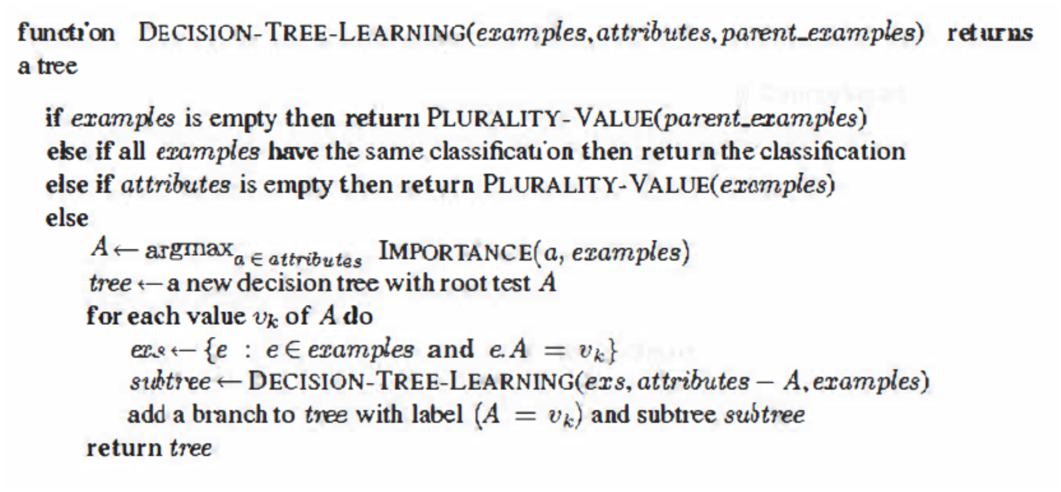
[1] Taken from [24], page 103.

Table 2.1: The dataset for the decision tree example[2]

Outlook	Temperature	Humidity	Wind	PlayTennis
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rain	Mild	High	Weak	Yes
Rain	Cool	Normal	Strong	Yes
Rain	Cool	Normal	Strong	No
Overcast	Cool	Normal	Strong	Yes
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Rain	Mild	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes
Rain	Mild	High	Strong	No

A decision tree is built from a set of training instances where each instance contains values for all of the attributes to be used in the decision tree along with the value of the attribute in question (in this case, whether or not to play tennis). This dataset can consist of any number of instances, though a larger dataset is much more likely to produce an accurate decision tree. The attributes are the potentially relevant features of the instances in the dataset; they are used to find the patterns, similarities and differences within the data that will help to classify a new instance. The decision tree in Figure 2.1 was ultimately constructed from the dataset shown in Table 2.1. This dataset consists of 14 total instances, with four predictor attributes and one class attribute. In this case, each attribute has a finite set of possible values. For example, the outlook attribute has three possible values: sunny, overcast and rain. As explained previously, the decision tree that is going to be constructed will be responsible for deciding whether or not a person should play tennis on that specific day.

[2] Taken from [24], page 10.



```
function DECISION-TREE-LEARNING(examples, attributes, parent_examples) returns
a tree
  if examples is empty then return PLURALITY-VALUE(parent_examples)
  else if all examples have the same classification then return the classification
  else if attributes is empty then return PLURALITY-VALUE(examples)
  else
      $A \leftarrow \text{argmax}_{a \in attributes}$ IMPORTANCE($a$, examples)
      tree ← a new decision tree with root test $A$
      for each value $v_k$ of $A$ do
          exs ← $\{e : e \in examples$ and $e.A = v_k\}$
          subtree ← DECISION-TREE-LEARNING(exs, attributes − $A$, examples)
          add a branch to tree with label $(A = v_k)$ and subtree subtree
      return tree
```

Figure 2.2: The general decision tree learning algorithm[3]

Figure 2.2 outlines the general decision tree learning algorithm that is used to create a decision tree from a given dataset of instances [?].

The first step to creating a decision tree involves reviewing each of the attributes and deciding which of them is the most important given the entire dataset. Once the most important attribute has been found, it is placed at the root, or top, of the decision tree. Each value in the selected attribute's domain then creates a stem from the root and each instance in the original dataset is pushed down the stem that corresponds to its value for that attribute. As a result, the dataset has been divided into a total number of n subsets, where n is the number of possible values corresponding to the selected attribute. At this point in the decision tree learning algorithm, this entire process is repeated with a few subtle differences. When an attribute is used as a splitting point in a decision tree, that attribute may not be used again within any of the sub-trees that it spawns. Therefore, there is no need to consider the importance of an attribute that has already been used above. In addition, only the relevant subset of instances, not the entire dataset, is used to calculate the importance of the attributes at any point in

[3] Taken from [21], page 702.

the tree. The decision tree construction process continues until either no attributes are left to split on or the importance value for each of the remaining attributes is 0. In the case that none of the remaining attributes achieves an importance value greater than 0, the majority class value of the relevant subset is used as the classification value.

In order to determine the importance of an attribute in the decision tree learning algorithm, the concepts of entropy and information gain are used. Entropy is a numerical value that is calculated using the subset of instances that have found their way down the tree to the node under review. This value reflects the impurity of the subset of data at this point. The more pure the subset, the lower the entropy value will be. More specifically, equation (2.1) shows the formal calculation. In the equation, S is the subset of data and p_i is the proportion of instances in the subset with a particular class value.

$$Entropy(S) = \sum_{i=1}^{n} -(p_i) * \log_2(p_i) \tag{2.1}$$

Using equation (2.1), the entropy of the original dataset, labeled as D, would be calculated as follows:

$$\begin{aligned} Entropy(D) &= (-\frac{5}{14} * \log_2 \frac{5}{14}) + (-\frac{9}{14} * \log_2 \frac{9}{14}) \\ &= .5305 + .4098 \\ &= .9403 \end{aligned}$$

Unlike entropy, information gain is a numerical value that is calculated using the remaining subset of data as well as a single attribute. This value reflects how useful the chosen attribute is in further splitting the instances into subsets that are pure, meaning that each subset is predominantly made up of instances with the same class value. The more useful an attribute proves to be, the higher its information gain value will be. Equation (2.2) shows how to calculate information gain. In the equation, S is the subset of instances, A is the attribute under consideration for splitting S further, v is a value of attribute A and S_v is the set of instances that have an attribute value of v for attribute A.

$$InformationGain(S, A) = Entropy(S) - \sum_{v \in A} \frac{|S_v|}{|S|} * Entropy(S_v) \tag{2.2}$$

Using equation (2.2), the information gain for splitting the original dataset D using the outlook attribute would be calculated as follows:

$$\begin{aligned} InformationGain(D, Outlook) &= .9403 - ((\frac{5}{14} * Entropy(D_{Sunny})) + \\ &\quad (\frac{4}{14} * Entropy(D_{Overcast})) + \\ &\quad (\frac{5}{14} * Entropy(D_{Rain}))) \end{aligned}$$

Using the concepts of information gain and entropy, the importance of an attribute may be calculated at any point in the decision tree. First, the entropy of the relevant dataset is calculated. Using this entropy value, different information gain values are derived for each of the relevant dataset's attributes. In the case of our decision tree algorithm, an attribute's information gain value is equal to its importance value. Therefore, the attribute that records the highest information gain value is used as the splitting attribute at that point in the decision tree. In Figure 2.1, the root node denotes that the initial split was made on the outlook attribute because it recorded the highest information gain value after reviewing the original dataset. This method of calculating an attribute's importance is always used to select which attribute, if any, should be used as the next splitting attribute in the decision tree creation process.

In evaluating a decision tree, one evaluates its accuracy on a subset of instances that were not used to construct the tree; this set is known as the test set. The accuracy of a decision tree is simply the number of instances from the test set that the tree classified correctly. Because it is often difficult to get data to train and test the decision tree on, a method called n-fold cross-validation is often used. In n-fold cross-validation, there are n total iterations, where the dataset is split into n equal-sized subsets. In a single iteration, one subset is removed from the dataset and placed in a testing set by itself, with the other (n-1) subsets combined to produce a training set. The training set is then used to form a decision tree, as done above, that will classify the instances in the testing set. This process is done for each of the subsets in the dataset and the overall accuracy is calculated by summing the number of correctly classified instances over the number of instances in the dataset. A special case of n-fold cross-validation is leave-one-out cross-validation, which is the act of performing n-fold cross-validation where n is equal to the number of instances in the dataset. In this scheme, a single instance will be tested against the decision tree constructed from the training set which contains all of the other instances. Much like n-fold cross-validation, this process repeats until each instance in the original dataset has been the sole instance of the testing set. When cross-validation is used, the overall accuracy is computed as the average accuracy of the n folds.

Cross-validation offers important advantages. One advantage is its ability to use as much data as possible for training, which is important for a small dataset. In addition, for a given value of n, n different training sets are constructed, each of which consists of a different subset of instances from the original dataset. Consequently, different decision trees are produced for each subset of instances due to changes in entropy and information gain values and each instance in the original dataset appears as part of the test set for one training iteration. If cross-validation techniques are not used and the overall dataset is partitioned into a single training set and testing set, there exists the potential risk of choosing training and testing sets that are skewed and result in an extremely high or low accuracy. Therefore, cross-validation techniques are often employed to find a realistic accuracy statistic. Once the method has been evaluated using cross-validation, the entire dataset is used to construct a model for subsequent use in classifying new instances.

The concept of a decision tree is very important because it will be the fundamental structure used to identify the requisite structure and content of graphs that are relevant to a user's query.

Chapter 3
DATA COLLECTION AND STORAGE

In order to begin examining how to extract the specified features from a user-written query, a corpus of queries must first be constructed. Two different human subject experiments were conducted to collect separate datasets composed of sample queries.

3.1 The First Human Subject Experiment

The first human subject experiment was administered to a random collection of University of Delaware undergraduates enrolled in a computer science course to collect an initial corpus of data for analysis. Each participant was at least 18 years of age and either a native or fluent English speaker. In this human subject experiment, the students were shown a single information-graphic and asked to enter a query that could be answered by the displayed graphic. A total of 20 sample information-graphics, one of which is shown in Figure 3.1, were used in the experiment, each was based on a different subject such as adoption, oil prices or television shows.

The students were given an hour to submit as many queries as possible (one per graphic) without sacrificing the quality of their entries. The first human subject experiment had a duration of approximately five weeks, ultimately collecting more than 350 queries from over 30 subjects. This corpus was intended to be used as the primary dataset for analysis and retrieval system evaluation. The first experiment was approved by the human subjects Review Board and each participant was given extra credit in his or her course for participation.

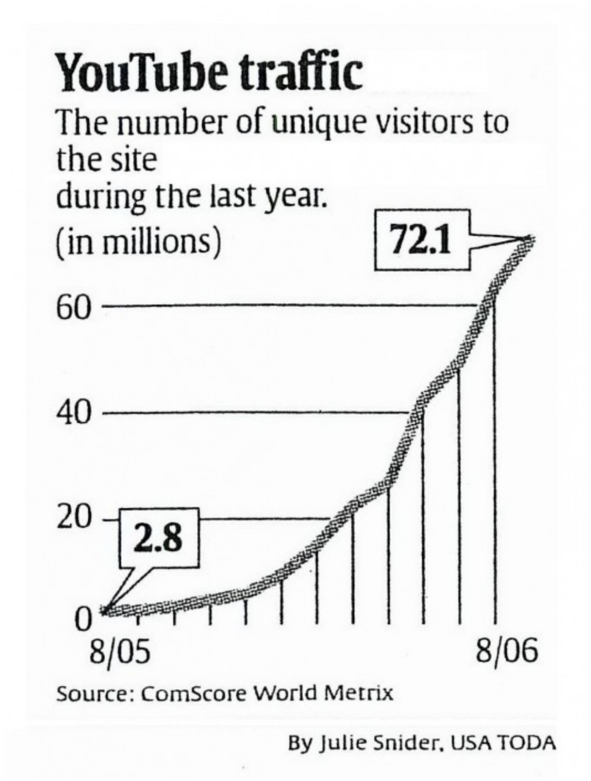


Figure 3.1: A sample information-graphic from the first experiment

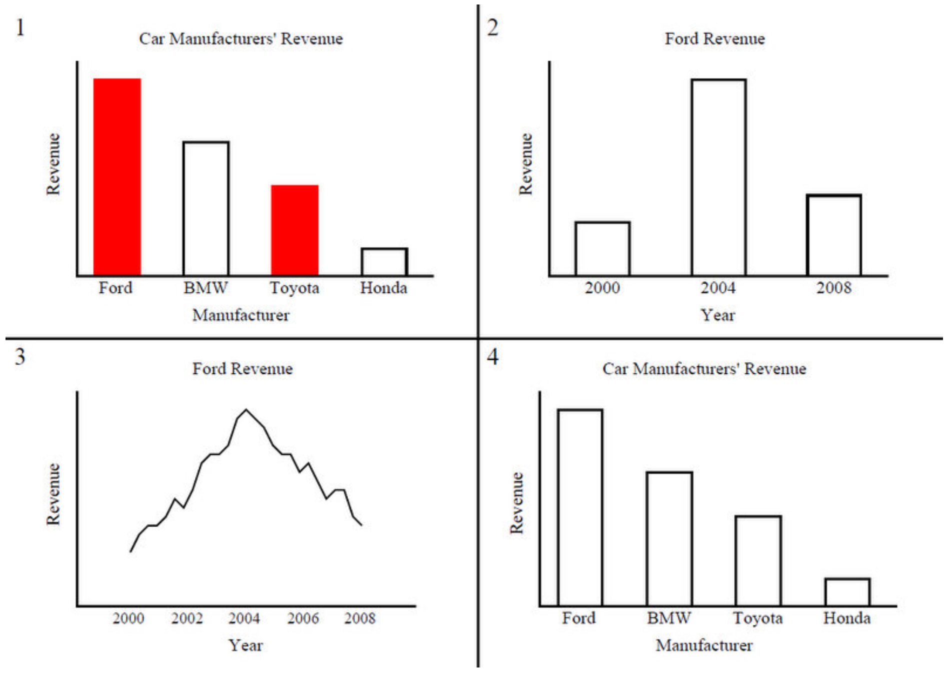


Figure 3.2: A sample information-graphic from the second experiment

3.2 The Second Human Subject Experiment

Much like the first human subject experiment, the second one was also administered to a random collection of University of Delaware undergraduates enrolled in a computer science course. Again, each participant was at least 18 years of age and either a native English speaker or fluent in English. In this human subject experiment, students were shown a set of four related information-graphics. Each graphic of a given set was constructed from related and sometimes identical data, though each of them often had a different high-level intended message. For example, one graphic might have conveyed the relative rank of a set of entities with respect to the value of some attribute, while another may have conveyed the trend of the value of a single entity's attribute over time. Figure 3.2 shows a sample set of information-graphics taken from this experiment.

The students were then asked to submit a query for each of the graphics, where the written query would be best answered by the chosen graphic as opposed to the others in the set. There were a total of five unique sets, based on various topics including car manufacturers, credit cards and mobile phone companies. Each student was given an hour to submit as many queries as possible (one per graphic, four per set) without sacrificing the quality of their entries. This experiment lasted about three weeks, successfully collecting over 250 queries from approximately 15 students. Like the first experiment, the second experiment was approved by the human subjects Review Board and each participant was given extra credit in his or her course for participation.

This experiment differs from the first in the fact that it shows multiple graphics at once where the domains of the graphics are similar or the same. This was done in order to force the human subjects to write queries that recognized subtle differences in the way that the data was displayed. For example, in Figure 3.2, the user had to write two unique queries for graphics number three and four. The differences in the two resulting queries may help to show linguistic features that differentiate queries written to return information-graphics trend and rank intended messages. As a result, the second experiment's queries were more useful than the first's when searching for features that indicate which information-graphic intended message a query seeks.

3.3 The Database

Each of the two previously mentioned experiments were web-based, meaning that the user was able to sit at a computer with internet access and submit queries that were immediately stored in a SQL database. Through the use of HTML and PHP scripts, public access to this database has been enabled through web browsers. Each dataset may be sorted by the graphic each query pertains to and, in the case of the second experiment, the set that each query belongs to. The following locations can be accessed to view the collected corpora:

First Human Subject Experiment:

www.eecis.udel.edu/ stagitis/ViewAll.php

Second Human Subject Experiment:

www.eecis.udel.edu/ stagitis/SE/ViewAllSets_Graphics.php

3.4 Data Correction

Our system requires that a query must be in full sentence form rather than a set of keywords because this research will attempt to use the query's words and grammatical structure to label words and phrases as independent axis content, dependent axis content or neither. With a simple keyword search, it would be impossible to do so since random words could appear in any number of sequences.

To focus on the methodology for identifying the characteristics of graphics that are relevant to a user's query, several further assumptions are made:

1) The query is grammatically correct

2) The query contains no word misspellings or abbreviations

3) The query is understandable

As will be explained shortly, the extraction of features from queries will require the use of a parser. Parsers require that their input be written in proper English due to their inability to effectively interpret slang and incorrect grammar. Due to the pipelined design of the retrieval method, this necessary precondition is expanded to encompass the entire system, meaning that all user queries must be written in proper English. As a result, it was necessary for a human to iterate through both corpuses to expand abbreviations and correct any minor spelling or fundamental English errors. It is envisioned that the retrieval system would employ grammar and spelling checkers and that a preprocessor would expand abbreviations to full words. Research work exists on these tasks and will not be considered further in this thesis [23] [25].

If an observed query was unable to be interpreted, was unrelated to the displayed graphic or would require major changes in its formulation, then the query was removed from the corpus. For example, query Q_6 was unable to be interpreted without major changes and therefore was removed from the corpus.

Q_6: *What is the difference between the amount of americans in millions uses Visa and MasterCard?*

In addition to these recurring linguistic errors, a query was also removed from the corpus if it was judged to be requesting a graphic that was anything other than a simple bar chart or single line graph. The main culprits of this issue were queries that sought grouped bar charts and multiple line graphs, which is partially due to their inclusion in the first experiment's set of information-graphics. The reason for removing such queries is that, as stated previously, this research only deals with the retrieval of simple bar charts and single line graphs. After removing queries that did not meet the previously mentioned standards, the corpus was left with over 324 total queries.

Chapter 4

BUILDING THE INDEPENDENT AXIS AND DEPENDENT AXIS MODEL

This chapter is concerned with the extraction of the desired independent axis content and dependent axis content from a user-written query. The goal of this process is to correctly select noun phrases from the query itself that accurately depict the content that would be featured on the axes of the ideal resulting information-graphic. Consider Q_7 below:

Q_7: Which first world countries have the largest GDP?

Ideally, the retrieval system would extract 'first world countries' as the desired content of the independent axis and 'GDP' as content of the dependent axis. The developed system considers the ordered set of words in a query one at a time and attempts to determine whether that set is associated with the dependent axis, independent axis or neither. Part of our work here included determining which attributes might be useful in making this decision.

4.1 Global and Local Attributes

Two different types of attributes appear useful for determining whether an entity in a query captures the content of the independent or dependent axes. The first of the two, global attributes, are attributes whose value is based on the query as a whole. An example of a global attribute is if the query is a *What-is* question, which would be true only if the query began with the phrase "What is". On the other hand, local attributes are attributes that depend on the specific entity under consideration and possibly that entity's role and position in the query. An example of a local attribute is whether or

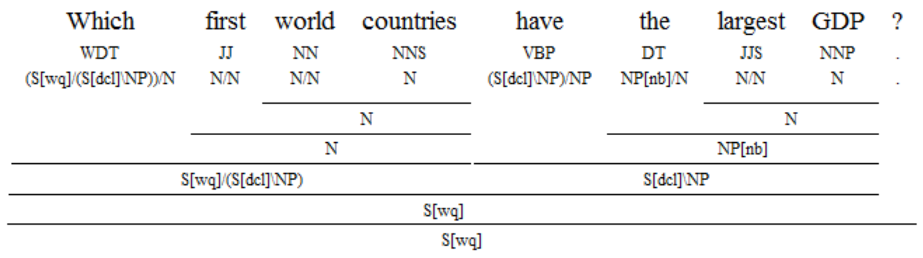

Which first world countries have the largest GDP ?
WDT JJ NN NNS VBP DT JJS NNP .
(S[wq]/(S[dcl]\NP))/N N/N N/N N (S[dcl]\NP)/NP NP[nb]/N N/N N .
N
N
N
NP[nb]
S[wq]/(S[dcl]\NP)
S[dcl]\NP
S[wq]
S[wq]

Figure 4.1: A sample query's parsing

not the entity is plural or singular. Classifying an entity is therefore dependent on both characteristics of the entity and the query containing the entity. The dataset consists of query-entity pairs; all of the entities from the same query have identical values for their global attributes.

4.2 Extracting Candidate Entities

In order to grasp the content of the desired information-graphic's independent and dependent axes, the system must first generate an exhaustive list of potential phrases from the query under review. To create this list of phrases, also known as candidate entities, each query is passed to Stephen Clark and James R. Curran's CCG parser [4]. The CCG parser is used to place part of speech tags on each of the words within the query and to help identify useful phrases. The parser displays the tagging in a tree structure to show where words come together to form phrases. The parse for Q_7 is shown in Figure 4.1. Once each of the words has been given a tag, the retrieval system selects phrases consisting of consecutive nouns and, if present, the leading noun's preceding adjective; it is common for these phrases to be broken apart by prepositions. This work was done by University of Delaware graduate student Zhuo Li. Query Q_7 spawns the following two candidate entities:

E_1*: first world countries*

E_2*: the largest GDP*

Once the list of candidate entities for a given query has been produced, each entity is matched with the query from which it was extracted to create a query-entity pair. A query-entity pair is comprised of a user-written query and an entity that was extracted from this same query. The following, denoted EP_1, illustrates a potential query-entity pair drawn from Q_7.

> EP_1: *[Which first world countries have the largest GDP?, first world countries]*

A dataset of 1072 query-entity pairs is created by cycling through the parsing process with each of the queries in the original corpus. It is important to note that the dataset of query-entity pairs is much larger than the dataset of queries since any given query typically spawned three to five candidate entities.

The goal is to label each query-entity pair in the dataset as one of the following three groups:

1) The independent axis descriptor phrase set
2) The dependent axis descriptor phrase set
3) The irrelevant phrase set

4.3 The Attribute Set

In order to build a successful decision tree for predicting an instance's class value, it is extremely important to first select attributes that will help to reveal patterns amongst the data set. A total of 42 attributes were developed in order to predict whether an entity captures the content of the independent axis, dependent axis, or neither of the two for graphs relevant to the user's query. All attributes, except for one, are binary with a value of 1 for true and 0 for false; the other attribute has three possible values. The following are examples of global attributes:

GA_1: *The query is a "What-is" question*

GA_2: *A trend word is contained within the query*

GA_3: *More than two entities are found in the candidate entity list*

The following are a few sample local attributes:

LA_1: *The entity is a time interval*

LA_2: *The entity is the left-most noun phrase in the query*

LA_3: *The entity follows a trend word in the query*

Global attribute GA_1 and local attribute LA_3 above both refer to the concept of a trend word. In the context of this research, a trend word is any word that falls within a specified, finite word set which we have found to discuss, describe or presuppose trends. This set is known as the trend word set. The trend word set contains words such as "changing", "increasing", "falling", and other words that would potentially describe the fluctuation of a value. A comparison word set, which contains words such as "rank", "versus" and "difference", is also used for similar purposes. The full set of attributes is shown in Table 4.1 and Table 4.2.

4.4 Annotating the Query-Entity Pairs

Zhuo Li has written scripts and functions that are able to accurately and effectively identify each of the attribute values for any given query-entity pair. Using these resources, it is not necessary for a human annotator to walk through the data set and record these values by hand. However, this was done early on in the process in order to check the accuracy of the automatic annotations. After all of the query-entity pairs' attribute values were identified, a human annotator was required to iterate through and label each of them with the correct judgment that should be given by the decision tree. The axis descriptor decision tree is then created and evaluated using the previously mentioned set of attributes and the human annotation as the correct classification.

4.5 Building the Decision Tree

Given the dataset of query-entity pairs, open-source data mining software known as WEKA [11] was used to construct the decision tree from the 42 attributes. The independent and dependent axes' decision tree takes a query-entity pair as input. From

Table 4.1: The attributes for the independent and dependent axes decision tree

Attribute	Description
superlative	A superlative adjective exists within the query
jjrlnSent	A comparative adjective exists within the query
trendWord	A word in the trend word set exists within the query
comparisonWord	A word in the comparison word set exists within the query
whatWhich	The query begins with "What" or "Which"
howManyMuch	The query begins with "How many" or "How much"
howDoHave	The query begins with "How do" or "How have"
whatIs	The query begins with "What is"
trendVerb	The main verb in the query is a verb from the trend word set
compareVerb	The main verb in the query is a verb from the comparison word set
compareHasC	The preposition is associated with the main comparison verb
compareNNS	The preposition associated with the main comparison verb compares one to one
compareNAll	The preposition associated with the main comparison verb compares one to many
compareNS	The preposition associated with the main comparison verb compares many
compareEOS	The main comparison verb is at the end of the query
manyFull	There are more than two noun phrases in the query
conjunction	There is a conjunction in the query
superlativeNP	This entity contains a superlative adjective
gradientNP	This entity contains a gradient word
quantitativeNP	This entity contain a quantitative word
comparisonNp	This entity contains a comparison word
trendNP	This entity contains a trend word
allPhraseNP	This entity contains an all-phrase
otherPhraseNP	This entity contains an other-phrase
eachPhraseNP	This entity contains an each-phrase
modifiedQuantNP	This entity is modified by a quantitative word
modifiedTrendNP	This entity is modified by a trend word
modifiedCompNP	This entity is modified by a comparison word
modifiedGradNP	This entity is modified by a gradient word

Table 4.2: Continuation of Table 4.1

Attribute	Description
whichWhatNP	This entity directly follows the “Which” or “What” query’s head
howManyMuchNP	This entity directly follows the “How many” or “How much” query’s head
howDoHaveNP	This entity directly follows the “How do” or “How have” query’s head
whatIsNP	This entity directly follows the “What is” query’s head
compBeforeNP	This entity appears before the main comparison verb
compAfterNP	This entity appears after the main comparison verb
trendBeforeNP	This entity appears before the main trend verb
trendAfterNP	This entity appears after the main trend verb
pluralNP	This entity is in plural form
complexNP	This entity is a complex noun phrase
headNP	This entity is the head noun of the query
fullNP	This entity is a noun phrase that is not part of a larger noun phrase
timeNP	This entity is a specific time interval, time point or neither

this pair, the tree will output ‘X-Helpful’ if the entity captures content on the independent axis, ‘Y-Helpful’ if it captures content on the dependent axis or ‘Not-Helpful’ if it does not capture content of either axis of potentially relevant graphs.

Chapter 5

EVALUATING THE INDEPENDENT AXIS AND DEPENDENT AXIS MODEL

5.1 Decision Tree Evaluation Method

Up to this point, the system takes a query as input and produces a list of the entities extracted from the query and whether the entity captures the content of the independent axis, dependent axis or neither. Whether or not a given entity captures this content is determined by traversing the decision tree using the attribute values computed for the given query-entity pair.

As discussed in Chapter 2, one common evaluation technique is leave-one-out cross-validation. However, the use of both global and local attributes causes an issue in implementing cross-validation techniques, especially leave-one-out cross-validation. Some of the attributes are global, which means that these attributes will have the same value for each query-entity pair belonging to the same query. Therefore, if only one query-entity pair is left out during a training run, the classifier will be trained with an unfair advantage. The unfair advantage occurs when both the training set and the testing set contain entities extracted from the same query, which can happen whenever a single query spawns multiple entities. This would mean that the same query, along with its identical global attribute values, would exist in both the training and testing sets at the same time, which is not ideal. In order to prevent this from happening, a modified version of leave-one-out cross-validation, which is referred to as leave-one-query-out cross-validation, was developed. In this procedure, all of the entities extracted from the same query will be bunched together and used as the testing set while all of the remaining entities will be used to form the decision tree. A total of n iterations of this custom cross-validation is performed, where n is the number of

Table 5.1: The confusion matrix for the axis decision tree

		System Output		
		'Not-Helpful'	'X-Helpful'	'Y-Helpful'
Human Annotation	'Not-Helpful'	43	9	24
	'X-Helpful'	5	325	86
	'Y-Helpful'	10	64	506

unique queries in the data set. Now, all query-entity pairs originating from the same query are guaranteed to be in either the training set or the testing set, but not both. Accuracy will be computed as the percentage of query-entity pairs that are correctly classified - that is, the result produced by the decision tree matches the classification assigned by the human annotator.

5.2 Baselines

After reviewing the dataset of query-entity pairs, examination of the human annotations reveals that 76 entities were labeled as 'Not-Helpful', 416 entities were labeled as 'X-Helpful' and 580 entities were labeled as 'Y-Helpful'. Clearly, the majority of extracted entities were found to be useful when describing the content of the dependent axis of potentially relevant information-graphics. As a result, the retrieval system could achieve an overall accuracy of 54% on the query-entity pair dataset by labeling all of the entities as 'Y-Helpful'. We therefore take this as our baseline accuracy.

5.3 Results

Using leave-one-query-out cross-validation, the system achieves an overall accuracy of approximately 81%. This is a huge improvement from the above baseline. Table 5.1 illustrates the confusion matrix for the output of the decision tree. A confusion matrix is a visual representation of the distribution of the classification of instances, with i rows representing the human annotations and j columns representing the system outputs. In particular, the entry in row i and column j shows how often the system classified an entity as the label at the top of column j when the correct classification was the label on row i. For example, the entry in row one and column two shows that nine instances were incorrectly classified by the system as 'X-Helpful' when the correct classification was 'Not-Helpful'. A confusion matrix shows where the system was correct and where the system made mistakes, which allows analysis to uncover where the decision tree had the most trouble. If we sum each entry for which i and j are equal, then we have the number of times the human and system classified an instance the same way. All numbers that do not fall on this diagonal represent instances that were given incorrect classifications by the decision tree.

From this confusion matrix, the precision and recall for both the independent axis and dependent axis can be calculated [15]. Precision refers to the fraction of the entities that the system identified as relevant to a given axis that were correctly classified. This statistic is important because it shows how often the system is correct when classifying an entity as part of the content of a given axis. On the other hand, recall refers to the fraction of the entities relevant to the axis being observed that are successfully identified. The recall statistic is also important because it reveals the proportion of total relevant entities that were found by the decision tree. The calculations for the precision and recall of the independent axis are shown below:

$$\begin{aligned} Precision_X &= \frac{325}{325+9+64} \\ &= .81658 \end{aligned}$$

$$\begin{aligned} Recall_X &= \frac{325}{325+5+86} \\ &= .78125 \end{aligned}$$

Likewise, the calculations for the dependent axis are shown below:

$$\begin{aligned} Precision_Y &= \frac{506}{506+24+86} \\ &= .82149 \end{aligned}$$

$$\begin{aligned} Recall_Y &= \frac{506}{506+10+64} \\ &= .87241 \end{aligned}$$

The precision and recall values for the independent axis, approximately 82% and 78% respectively, show that the decision tree is doing very well when deciding which entities should be accepted as independent axis content. Likewise, the precision and recall values for the dependent axis, both of which are also above 80%, show that the decision tree is able to accurately differentiate dependent axis content from non-dependent axis content. The exceptionally high recall value of about 87% for the dependent axis means that the decision tree excels the most when locating all of the entities that convey information belonging on that axis. After analyzing the confusion matrix, it is easy to see that the majority of the decision tree's errors occur when classifying an entity as dependent axis content when it is really independent axis content and vice versa.

Chapter 6
BUILDING THE INTENDED MESSAGE MODEL

Since this research is concerned with information-graphics that appear in popular media, the graphics themselves generally have a high-level message that they are intended to convey. For example, consider the graphic about credit card companies shown in Figure 6.1. This graphic is ostensibly intended to convey the high-level message that American Express ranks fourth, or second from last, among the companies listed in terms of the number of credit cards in circulation in 2003. The rank intended message is one of eight different categories that will be discussed in section 6.1 below.

For the system to successfully identify the intended message of a graphic, it must rely on the presence of communicative signals in the desired graphic. For example, salience is one kind of communicative signal; a bar becomes salient if it is colored differently from other bars or is mentioned within the graphic's caption. Either of these occurrences would suggest that the salient bar plays a major role in the graphic's high-level message. Salience is one of the features that has been used in a Bayesian network that hypothesizes an information-graphic's intended message [7].

Earlier on, a corpus of queries was collected to build a decision tree for identifying phrases in a query that capture the content of the axes of information-graphics that are potentially relevant to a user's query. This query dataset will also be used to construct a decision tree for identifying the category of intended message for potentially relevant graphics. However, a human annotator first examined the corpus of queries, labeling each of them with their intended message so that the dataset could be used for training and testing.

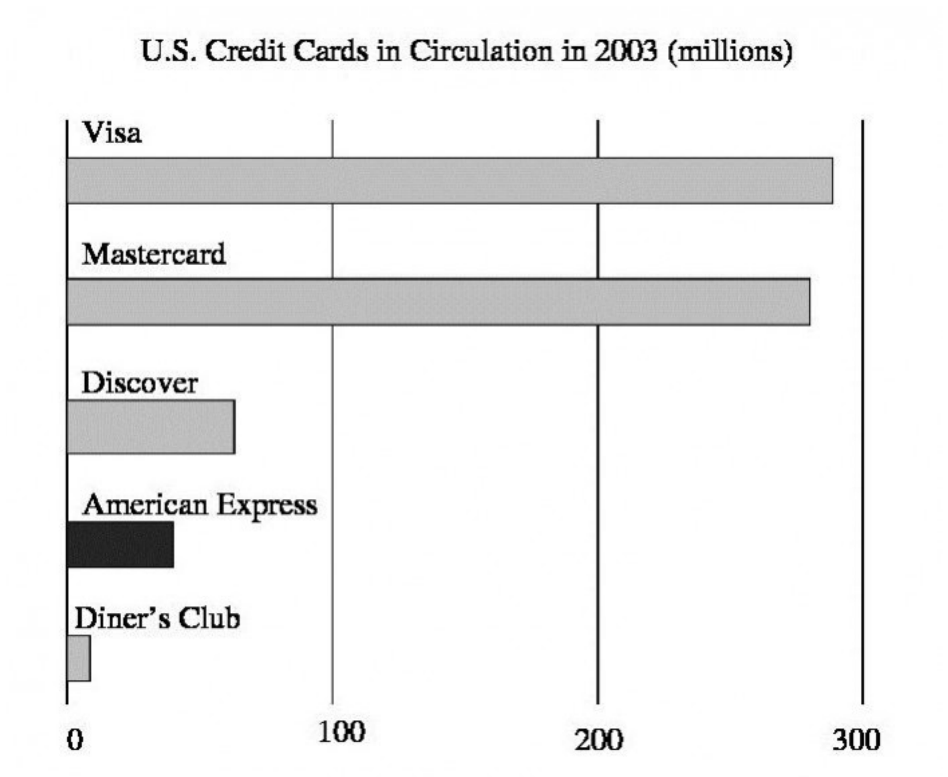


Figure 6.1: A sample information-graphic showing the total number of credit cards in circulation for various companies

6.1 The Intended Message Categories

Although a total of 22 different message categories were used in the systems for recognizing the intended message of bar charts and line graphs [6] [27], only eight condensed categories are useful to identify from users' queries. The reason that the original 22 were meshed together to form eight is that the original categories captured more detailed knowledge regarding the information-graphic itself than was likely to be found in a user query. For example, three of the original intended message categories were:

1) Stable trend message

2) Increasing trend message

3) Decreasing trend message

In this research, users are searching for relevant information-graphics that are unknown to them at the searching stage, which means that they are unlikely to differentiate among the three categories listed above. If a user is searching for the trend of the United States' GDP over the last ten years, he or she is unlikely to construct a query that specifies whether said trend is increasing, decreasing, or remaining stable. As a result, the three intended message categories are combined into a single trend category. Multiple combinations such as this are responsible for reducing the total number of categories from 22 to eight. Each intended message category and its description may be found in Table 6.1.

6.2 Using the Independent/Dependent Axis Content

The content of the independent axis and dependent axis play significant roles in identifying the category of intended message of a graphic, which is why this content is extracted first in the pipelined procedure. For example, if the query indicates that the independent axis should be a time interval, then the intended message of relevant graphics is likely to fall into the trend category. Similarly, the number of entities depicted on the independent axis is a clue about the intended message of relevant graphics. Consider the following example queries:

Table 6.1: The intended message categories

Intended Message Category	Description
Rank-All	A set of entities are being ranked against one another
Rank	A single entity is being ranked against a set of other entities of the same ontology
Relative-Difference	Two entities are being compared with one another
Single Maximum/Minimum	A single entity with the highest or lowest attribute value is identified
Multiple Maximum/Minimum	A set of entities with the highest or lowest attribute values are identified
Trend	An attribute of an entity is being conveyed over an ordinal period, usually time
Single General	General information regarding a single entity
Multiple General	General information regarding a set of entities

Q_8: *How does the revenue of Google rank among all technology companies?*

Q_9: *How does the revenue of Google compare with Facebook?*

Knowing that *Google* and *technology companies* capture content of the independent axis, and that one is singular while the other is plural, suggests that query Q_8 might be asking for a graphic of the rank category. On the other hand, knowing that Google and Facebook are both singular entities on the independent axis suggests that query Q_9 falls into the relative difference category. Although an information-graphic that includes the revenue of many technology companies, including *Google* and *Facebook*, could provide the information requested by query Q_9, the user can extract that information more easily from a graphic specifically devoted to Google and Facebook with no other distracting entities. Thus attributes based on the identified content of the axes include the number of independent axis entities (1, 2, more than 2), and their plurality (all singular, all plural, mixture of singular and plural). These are two attributes that help make up the attribute set for identifying the intended message.

Table 6.2: The attributes for the intended message decision tree

Attribute	Description
num-of-X	The number of entities that were classified as independent axis content
num-of-Y	The number of entities that were classified as dependent axis content
singular-plural-X	Whether all independent axis content entities are singular, plural, or a mixture of the two
superlatives	Whether or not a superlative exists in the query
verb-type	Whether the main verb in the query is a trend verb, comparison verb or neither
NP-relation	Specifies the singular/plural relationship between the independent axis entities
time-interval	Whether or not a time interval, time point or neither exist in the query
modified-X	Whether or not phrases such as “all”, “other” and “each” modify the independent axis entities

6.3 The Attribute Set

Along with the aforementioned two attributes, six other attributes are used to construct the decision tree for hypothesizing the category of intended message of potentially relevant graphics. One example of such attributes is whether or not a time interval exists within the query. If a time interval is present in the query, then it is more likely that the query is requesting graphics whose intended messages fall into the trend category. Another example of an attribute is whether or not a special verb, such as a comparison or trend verb, is in the query. The existence of such verbs helps to provide insight into the query's intended message by separating the comparison queries from the trend queries. An exhaustive list of attributes and their descriptions is given in Table 6.2.

Once again, University of Delaware graduate student Zhuo Li has written functions that are able to identify the values for each of these attributes given a single user written query.

6.4 Building the Decision Tree

WEKA [11] was again used to construct a decision tree from the data set of queries. This decision tree, however, had a series of eight possible outputs, one for each of the intended message categories. Once the tree is built, the category of intended message for graphics relevant to a new query can be determined.

Chapter 7

EVALUATING THE MODEL FOR HYPOTHESIZING REQUISITE INTENDED MESSAGE

Leave-one-out cross-validation was used to evaluate the decision tree for hypothesizing the category of intended message of graphics that might be relevant to a user's query. Accuracy was again measured as the percentage of queries for which the system output matched the intended message assigned by the human annotator. Unlike the axis content data set, the baseline for the intended message category data set turns out to be slightly higher, approximately 58%. This is due to the large number of queries that are annotated as requesting information-graphics whose intended message falls into the trend category, which total 189. This baseline means that the retrieval system would achieve an overall success rate of 58% on the original data set if the trend intended message category was output for every query.

7.1 Results

After extracting the 8 attributes and constructing a decision tree to decide which intended message category is requested by a given query, the system achieves an overall accuracy of approximately 81% using leave-one-out cross-validation. Once again, this is a large improvement from the original baseline even though the baseline for this data set is larger than the previous one.

Table 7.1 illustrates the confusion matrix for the output of the decision tree. From this confusion matrix, the precision and recall for each of the intended message categories may be calculated. The results of the calculations are displayed in Table 7.2.

The precision and recall values when identifying queries that desire information-graphics with singular-maximum/minimum, multiple-maximum/minimum and trend

Table 7.1: The confusion matrix for the intended message category decision tree

		System Output							
		Rank-All	Rank	Relative-Differenc'	Single-Max./Min.	Multiple-Max./Min.	Trend	Single-General	Multiple-General
Human Annotation	Rank-All	11	2	3	0	0	3	0	0
	Rank	2	20	6	0	1	1	0	0
	Relative-Difference	0	2	13	0	0	2	0	0
	Single-Max./Min.	0	0	0	16	0	2	0	0
	Multiple-Max./Min.	0	2	0	0	17	2	0	0
	Trend	1	0	2	0	0	183	0	3
	Single-General	0	0	0	0	0	16	0	1
	Multiple-General	0	0	0	0	0	9	0	4

Table 7.2: The precision and recall values for each of the intended message categories

		Statistics	
		Precision	Recall
Hand Annotation	Rank-All	.786	.579
	Rank	.769	.667
	Relative-Difference	.542	.765
	Single-Max./Min.	1	.889
	Multiple-Max./Min.	.944	.81
	Trend	.84	.968
	Single-General	0	0
	Multiple-General	.5	.308

intended message categories were all above 80%. The decision tree does very poorly in successfully locating queries that desire graphics with single-general and multiple-general intended message categories. The precision and recall values for each of these categories are very low, most less than or equal to 50%. This poor performance means that the decision tree is unable to consistently separate these types of queries from the rest of the ones in the dataset. The decision tree was able to label the queries with relative-difference, rank-all and rank intended message categories fairly well, achieving precision and recall values that fell between 54% and 79%. The confusion matrix shows in greater detail the decision tree's failure to identify queries belonging to the general intended message categories.

Chapter 8
CONCLUSION AND FUTURE WORK

8.1 Conclusion

Image Meta search and Content-based Image Retrieval methods have had some success for major search engines when attempting to retrieve pictorial images based on a user's query. When searching for information-graphics, however, these techniques fall short because they do not attempt to utilize the actual structure and high-level message of the information graphic.

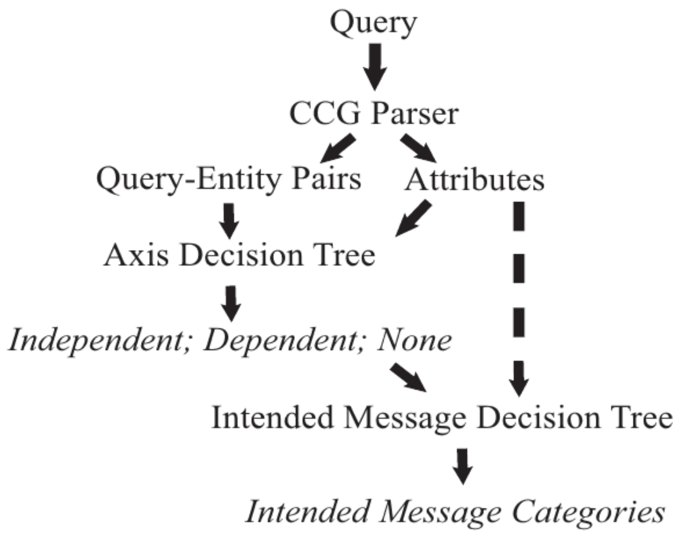

Query
CCG Parser
Query-Entity Pairs
Attributes
Axis Decision Tree
Independent; Dependent; None
Intended Message Decision Tree
Intended Message Categories

Figure 8.1: The overall method for the information extraction system

This thesis introduces the first steps of a new pipelined design, as illustrated in Figure 8.1, that has an approximately 80% success rate when identifying the content of the axes and the intended message category of graphs that are potentially relevant to a user's query. By using this research, a search engine will be able to use additional information when ranking potential results against one another, therefore hopefully yielding a more effective search for relevant graphics.

8.2 Future Work

Although the two decision trees have performed well above their respective baselines, improvements are always possible. One such improvement is the development of additional attributes for both the independent axis and dependent axis decision tree and the intended message category decision tree. Another improvement that could be made lies with the parser that is used to extract the candidate entities for the axis decision tree. With the current system, the parser oftentimes selects phrases that contain words of little value when attempting to express the content of the independent and dependent axes. Post-processing of the candidate entities could help to eliminate any such words so that the content of each axis is more easily understood. Finally, our dataset is relatively small and a larger dataset could lead to better results.

BIBLIOGRAPHY

[1] Youssef Bassil. A survey on information retrieval, text categorization, and web crawling. *arXiv preprint arXiv:1212.2065*, 2012.

[2] Ana B Benitez, Mandis Beigi, and Shih-Fu Chang. A content-based image meta-search engine using relevance feedback. *IEEE Internet Computing*, 2(4):59–69, 1998.

[3] Sandra Carberry, Stephanie Elzer, and Seniz Demir. Information graphics: an untapped resource for digital libraries. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 581–588. ACM, 2006.

[4] Stephen Clark and James R Curran. Wide-coverage efficient statistical parsing with ccg and log-linear models. *Computational Linguistics*, 33(4):493–552, 2007.

[5] Ritendra Datta, Dhiraj Joshi, Jia Li, and James Z Wang. Image retrieval: Ideas, influences, and trends of the new age. *ACM Computing Surveys (CSUR)*, 40(2):5, 2008.

[6] Stephanie Elzer, Sandra Carberry, and Ingrid Zukerman. The automated understanding of simple bar charts. *Artificial Intelligence*, 175(2):526–555, 2011.

[7] Stephanie Elzer, Sandra Carberry, Ingrid Zukerman, Daniel Chester, Nancy Green, and Seniz Demir. A probabilistic framework for recognizing intention in information graphics. In *International Joint Conference On Artificial Intelligence*, volume 19, page 1042. Lawrence Erlbaum Associates LTD, 2005.

[8] Claire Fautsch and Jacques Savoy. Adapting the tf idf vector-space model to domain specific information retrieval. In *Proceedings of the 2010 ACM Symposium on Applied Computing*, pages 1708–1712. ACM, 2010.

[9] Massimo Franceschet. Pagerank: Standing on the shoulders of giants. *Communications of the ACM*, 54(6):92–101, 2011.

[10] James L Hieronymus. Ascii phonetic symbols for the worlds languages: Worldbet. *Journal of the International Phonetic Association*, 23, 1993.

[11] Geoffrey Holmes, Andrew Donkin, and Ian H Witten. Weka: A machine learning workbench. In *Intelligent Information Systems, 1994. Proceedings of the 1994 Second Australian and New Zealand Conference on*, pages 357–361. IEEE, 1994.

[12] Farshad Kyoomarsi, Hamid Khosravi, Esfandiar Eslami, Pooya Khosravyan Dehkordy, and Asghar Tajoddin. Optimizing text summarization based on fuzzy logic. In *Computer and Information Science, 2008. ICIS 08. Seventh IEEE/ACIS International Conference on*, pages 347–352. IEEE, 2008.

[13] Joon Ho Lee. Properties of extended boolean models in information retrieval. In *SIGIR94*, pages 182–190. Springer, 1994.

[14] Ying Liu, Dengsheng Zhang, Guojun Lu, and Wei-Ying Ma. A survey of content-based image retrieval with high-level semantics. *Pattern Recognition*, 40(1):262–282, 2007.

[15] Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to information retrieval*, volume 1. Cambridge University Press Cambridge, 2008.

[16] David James Miller. Systems and methods for document searching, February 23 2012. US Patent App. 13/403,268.

[17] M Narayana and Subhash Kulkarni. Content based image retrieval using sketches. In *Proceedings of International Conference on Advances in Computing*, pages 1117–1123. Springer, 2012.

[18] Wei Peng, Juhua Chen, and Haiping Zhou. An implementation of id3—decision tree learning algorithm. *From web. arch. usyd. edu. au/wpeng/DecisionTree2. pdf Retrieved date: May*, 13, 2009.

[19] Vijay V Raghavan and SKM Wong. A critical analysis of vector space model for information retrieval. *Journal of the American Society for information Science*, 37(5):279–287, 1986.

[20] Stephen Robertson and Hugo Zaragoza. *The Probabilistic Relevance Framework*, volume 3. Now Publishers Inc, 2009.

[21] Peter Norvig Stuart J. Russell. *Artificial intelligence: a modern approach.* Prentice Hall Englewood Cliffs, third edition, 1995.

[22] Quan Wang, Jun Xu, Hang Li, and Nick Craswell. Regularized latent semantic indexing: A new approach to large-scale topic modeling. *ACM Transactions on Information Systems (TOIS)*, 31(1):5, 2013.

[23] Eric Watson, Marcelo Calbucci, Sally Salas, and Darren Shakib. Query preprocessing and pipelining, December 30 2008. US Patent 7,472,113.

[24] Ian H Witten and Eibe Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2011.

[25] Wilson Wong, Wei Liu, and Mohammed Bennamoun. Integrated scoring for spelling error correction, abbreviation expansion and case restoration in dirty text. In *Proceedings of the fifth Australasian conference on Data mining and analystics-Volume 61*, pages 83–89. Australian Computer Society, Inc., 2006.

[26] Ho Chung Wu, Robert Wing Pong Luk, Kam Fai Wong, and Kui Lam Kwok. Interpreting tf-idf term weights as making relevance decisions. *ACM Transactions on Information Systems (TOIS)*, 26(3):13, 2008.

[27] Peng Wu, Sandra Carberry, Stephanie Elzer, and Daniel Chester. Recognizing the intended message of line graphs. In *Diagrammatic Representation and Inference*, pages 220–234. Springer, 2010.